PLATAFORMA WEB PARA ADMINISTRADOR DE EVENTOS PARA LA COMUNIDAD UNIVERSITARIA

BRAYNT ROLANDO GUERRERO FLÓREZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FISICOMECANICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2019

PLATAFORMA WEB PARA ADMINISTRADOR DE EVENTOS PARA LA COMUNIDAD UNIVERSITARIA

BRAYNT ROLANDO GUERRERO FLOREZ

Trabajo de Grado para Optar el Título de Ingeniero de Sistemas e Informática

DIRECTOR
GABRIEL RODRIGO PEDRAZA FERREIRA
Doctorado en Ciencias de la Computación

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FISICOMECANICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2019

CONTENIDO

LISTA DE TABLAS

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE ANEXOS

Pág.

RESUMEN

TÍTULO: PLATAFORMA WEB PARA ADMINISTRADOR DE EVENTOS PARA LA COMUNIDAD UNIVERSITARIA.*

AUTOR: BRAYNT ROLANDO GUERRERO FLOREZ**

PALABRAS CLAVE: API, REST, plataforma, evento, JSON, Smart Campus, Smart Cities.

DESCRIPCIÓN:
La Universidad Industrial de Santander es una de las universidades más importantes sobresalientes del oriente colombiano y por lo tanto también del país. Es bien conocida en la ciudad de Bucaramanga y en general en toda la región del oriente porque la mayoría de eventos importantes destacados se llevan a cabo allí. Por Además de ser un importante centro del conocimiento, y de estar está compuesta por una comunidad tan bastante grande, entre los cuales se cuentan estudiantes, docentes y administrativos. La realización de eventos es una de sus facetas fundamentales significativas que fomentan su desarrollo como organización.

Este proyecto pretende abarcar la problemática que ha surgido en la universidad y centralizar la publicación y difusión de eventos en el campus usando adecuadamente la información que se genera en este ejercicio, y de esta manera intentar tener un impacto positivo en la calidad de vida de la comunidad universitaria.

Quick Events nace a raíz de esta dificultad y va especialmente dirigida al usuario que organiza y dirige eventos, para que pueda darlos a conocer a toda la comunidad y al mismo tiempo obtener una serie de resultados acorde a sus publicaciones para determinar qué tipo de eventos gustan a la comunidad e impulsarlos aún más.

El presente proyecto es una manera que tiene la universidad de impulsar la iniciativa Smart Campus.

* Trabajo de Grado
** Facultad de Ingenierías Fisicomecanicas. Escuela de Ingeniería de Sistemas e Informática. Director: Gabriel Rodrigo Pedraza Ferreira, Doctorado en Ciencias de la Computación.

ABSTRACT

TITLE: WEB PLATFORM FOR EVENT MANAGER FOR THE UNIVERSITY COMMUNITY*

AUTHOR: BRAYNT ROLANDO GUERRERO FLOREZ**

KEYWORDS: API, REST, platform, event, JSON, Smart Campus, Smart Cities.

DESCRIPTION:
The Industrial University of Santander is one of the most important universities in eastern Colombia and therefore in the country. It is well known in the city of Bucaramanga and in general throughout the eastern region because most of the important events take place there. Being an important focus of knowledge and being composed of such a large community, which includes students, professors and employees, the realization of events is one of its fundamental facets and that encourage its development as an organization.

This project aims to cover the problem that has arisen in the university and centralize the publication and dissemination of events on campus using the information generated in this exercise, and thus try to have a positive impact on the quality of life of the university community.

Quick events was born as a result of this difficulty and is specifically aimed at the user who organizes and directs events, so that they can make them known to the entire community and at the same time obtain a series of results according to their publications to determine what type of events they like to the community and push them further.

This project is a way for the university to promote the Smart Campus initiative.

* Bachelor Thesis

** Facultad de Ingenierías Fisicomecanicas. Escuela de Ingeniería de Sistemas e Informática. Director: Gabriel Rodrigo Pedraza Ferreira, Doctorado en Ciencias de la Computación.

INTRODUCCIÓN

En todo tipo de organizaciones existe un flujo constante de la información con diferentes objetivos y cada organización por lo general cuenta con los canales necesarios para su difusión, por ejemplo, en las empresas cuando se desea socializar un nuevo procedimiento es necesario informar a los trabajadores de dicho arreglo lo cual se puede hacer mediante correos internos, tableros de anuncios, carteles, perifoneo, entre otros.

En organizaciones como la Universidad Industrial de Santander se realizan eventos sobre diferentes temáticas, cuando esto ocurre, esta información generalmente se difunde por plataformas conocidas por los estudiantes de la Escuela de Ingeniería de Sistemas e Informática como *Cormorán (Comunidad Académica, COMA)* que permite el envío de correos masivos, creación de encuestas, eventos, y noticias. En otras ocasiones se disponen de carteles publicitarios distribuidos a lo largo del campus e incluso también a través de la página principal de la UIS y por sus cuentas oficiales en redes sociales.

El número anual de eventos y presentaciones culturales internacionales realizados en la UIS en el año 2016 y 2017 fue de 23 y 30 respectivamente. El número anual ponderado de participantes en eventos deportivos y lúdico-recreativos en el año 2016 y 2017 fue de 2127 y 1394 respectivamente. Así mismo en estos dos años el respectivo número anual de asistentes a actividades culturales en la Universidad Industrial de Santander fue de 102344 y 84615. Finalmente, en el año 2017 se realizaron 282 actividades culturales dirigidas a la comunidad universitaria.

Además, es interesante tener en cuenta que en el año 2017 en el primero y segundo semestre llegaron a estar matriculados en la sede Bucaramanga 17952 y 18597 estudiantes respectivamente en todos los niveles de formación. Por otra parte en

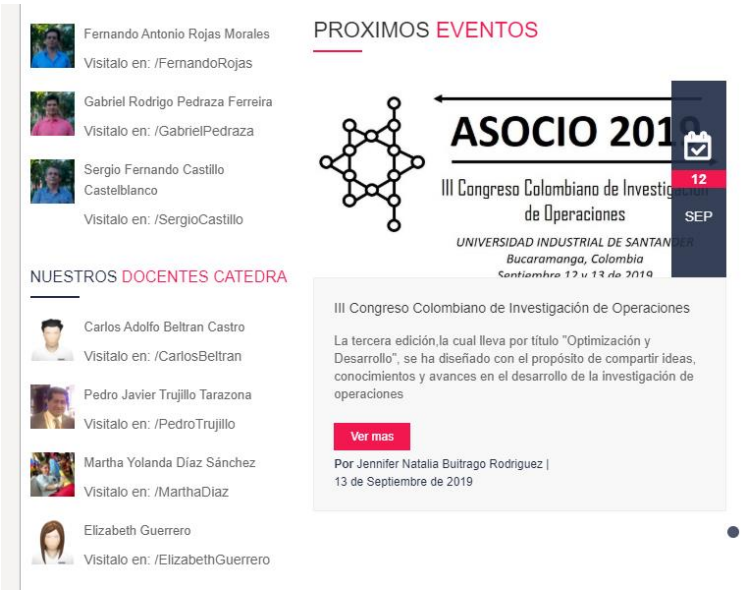
ese mismo año, en el segundo semestre se contaron 1757 docentes en total, y 1265 administrativos en todos los niveles de formación[1].

Los datos anteriores nos permiten tener una idea de cuanta afluencia de asistentes hay en la universidad en el desarrollo de eventos durante el año y la cantidad de estos que sea organizan cada año. Podemos concluir que es una buena cantidad.

De todos estos datos se concluye que en la UIS no solo se ofrecen buena cantidad y diversidad de eventos, sino también hay gran cantidad de posibles personas interesadas. El hecho de que la universidad organice eventos cada año genera información que sería útil manejar.

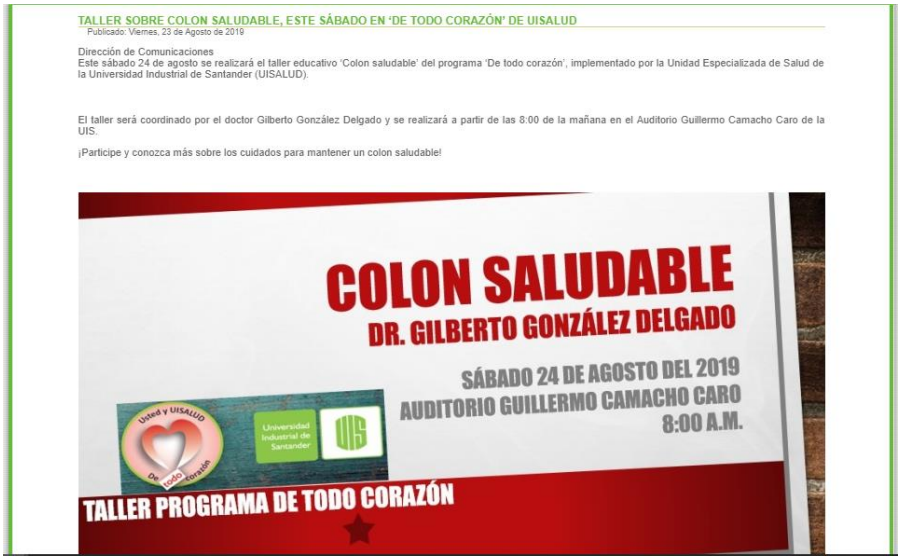
[1] UIS EN CIFRAS 2017. [En línea]. (Recuperado en 30 Julio 2019). Disponible en https://usedsources.page.link/tc4X

Figura 1. Página de inicio – CORMORAN



Para conocer esta información es necesario que el estudiante este registrado en la plataforma, pero desafortunadamente hay escuelas en la universidad en la que se usa muy poco y muchos estudiantes aún no están registrados.

Figura 2. Evento en la página de Inicio - UIS



La página institucional constantemente se va actualizando, allí se publican eventos y noticias importantes que ocurren en el campus y también fuera de él. Estar al tanto

de este contenido requiere el hábito de consultar la página frecuentemente, pero esto es algo a lo que muchas personas no están dispuestas. Ciertamente es una desventaja.

Figura 3. Tablero de anuncios – CEIS

La figura 3 corresponde al tablero de anuncios que se encuentra en el Centro de Estudios de Ingeniería de Sistemas. En este lugar suelen reunirse los estudiantes para realizar trabajos y tareas además de esparcimiento.

Con este tipo de carteles quienes coinciden en este lugar tienen la posibilidad de enterarse de los eventos que se realizan. Sin embargo, no todos los estudiantes de la escuela se reúnen allí, muchos incluso a lo largo de su carrera nunca han entrado.

El problema entonces que se ha percibido es que este flujo de información al estar distribuido en diferentes canales no se cuenta con uno común y unificado donde, no solo los estudiantes, sino las demás personas que componen la comunidad universitaria conozcan oportuna y fácilmente qué novedades de eventos se están realizando en la universidad a fin de que haya un aprovechamiento más pleno de los servicios que esta ofrece.

Al abordar este problema es donde se desarrolla el presente proyecto como una posible solución que permita centralizar este flujo de información permitiendo su uso de manera fácil y rápida.

El proyecto está enfocado entonces al desarrollo de una plataforma web pensada desde el punto de vista de un administrador de eventos que le permita publicarlos y poder hacer un seguimiento de los eventos que va publicando, y así lograr cierta retroalimentación respecto a la actividad de negocio que se está llevando a cabo.
Para dicho desarrollo se ha planteado el uso de la metodología de desarrollo por prototipos. La idea es realizar dos prototipos donde el segundo será el final y definitivo; cada prototipo cuenta con ligeras diferencias, mejoras introducidas hasta desembocar en el prototipo final a fin de obtener una plataforma con funcionalidades básicas para acometer la solución del problema.

1. OBJETIVOS

1.1 OBJETIVO GENERAL

Crear plataforma web para administrar eventos de interés común para la comunidad universitaria de manera eficiente, y rápida.

1.2 OBJETIVOS ESPECÍFICOS

- Identificar los conceptos implicados en la realización de un evento y el flujo de información en una organización para así contextualizar el propósito de la aplicación a realizar.
- Proponer una arquitectura para la plataforma que la dote de propiedades como escalabilidad y usabilidad.
- Elaborar e implementar un primer prototipo funcional basado en la arquitectura mencionada anteriormente.
- Validar el funcionamiento de dicho prototipo de la plataforma, a través de un conjunto de pruebas funcionales.

2. MARCO REFERENCIAL

2.1 SMART CAMPUS

2.1.1 Smart City. El término "Smart City" se le atañe a una ciudad que hace uso de tecnologías de información y comunicación (TIC) en su gestión interna. Este uso tiene como objetivo mejorar la toma de decisiones, eficiencia de operaciones y mejorar la detección de necesidades sociales y ambientales. Esto promueve un desarrollo integral y sostenible que puede tener un impacto positivo en la calidad de vida de sus habitantes, también un espacio más seguro para la generación de empleo y reducir las desigualdades. Poco a poco esto se convierte en un ciclo que garantiza bienestar económico y social.

En nuestro país, Colombia, esta tendencia ha despertado intereses de altas esferas, que, en ciudades grandes como Medellín, Bogotá y Barranquilla, se le está dando más impulso, aunque como estos expertos admiten, es algo que toma tiempo y esfuerzo[2].

[2]ASÍ VA LA IMPLEMENTACIÓN DE LAS 'SMART CITIES' EN COLOMBIA. [En línea]. (Recuperado en 30 Julio 2019). Disponible en https://usedsources.page.link/GLp8

Figura 4. Maquina Recarga Verde

Fuente: LA RECARGA VERDE VUELVE CON MÁS MÁQUINAS [En línea]. (Recuperado en 25 de Agosto 2019). Disponible en https://usedsources.page.link/82AY

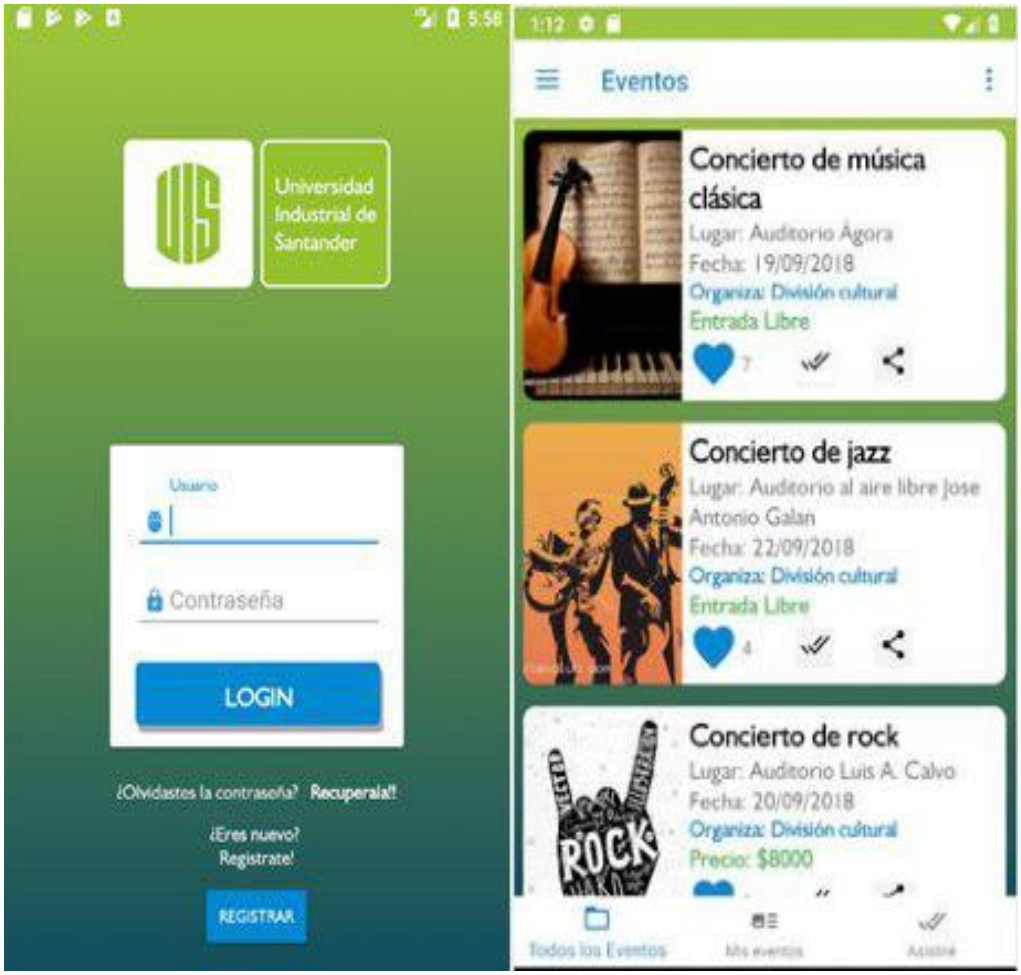
En la figura 4 se puede ver un ejemplo de una iniciativa Smart City que se está aplicando en la ciudad de Medellín. Allí vemos una máquina de Recarga Verde, la cual estimula el reciclaje de envases de plástico, aluminio y vidrio entre las personas mediante una recompensa en dinero que se refleja en la tarjeta cívica, tarjeta que se usa para acceder al servicio de transporte masivo de Medellín.

Los beneficios que ofrece este tipo de iniciativa ciertamente son considerables teniendo en cuenta que son millones los usuarios del transporte masivo de Medellín en sus diferentes líneas.

2.1.2 Concepto Smart Campus. El concepto ‘Smart’ puede ser trasladado a algo más pequeño que una ciudad. Así surge el término ‘Smart Campus’.

Un Campus inteligente (Smart Campus) es aquel que es capaz de aprovechar los datos que produce en su funcionamiento diario para generar información nueva que le permita mejorar su gestión y ser más sostenible, más competitivo y ofrecer mejor calidad de servicio y bienestar, gracias a la participación y colaboración de toda la comunidad universitaria.[3]

Figura 5. Componente aplicación móvil



Fuente: DURAN, Javier; DISEÑO DE UNA APLICACIÓN MOVIL DE DIFUSION E INTERACCION DE EVENTOS EN UNA INICIATIVA SMART-CAMPUS PARA LA UNIVERSIDAD INDUSTRIAL DE SANTANDER, 2019. p.49.

[3] MAZA FIGUEROA, Natalí; OROZCO ACEVEDO Melissa. MODELO DE GESTIÓN ESTRATÉGICA PARA EL DESARROLLO DE UN CAMPUS INTELIGENTE BASADO EN CONCEPTOS DE SMART CITY EN LA UNIVERSIDAD DE CARTAGENA – CAMPUS PIEDRA DE BOLÍVAR, 2017. p.39.

La anterior figura pertenece a la Aplicación móvil desarrollada por el estudiante Javier Fernando Durán que corresponde al usuario asistente a los eventos y complementa este proyecto WEB.

Teniendo en cuenta los antecedentes anteriormente mencionados ha nacido este proyecto, y es parte de un macroproyecto el cual se complementa con una aplicación móvil y un servicio backend que ofrece una API REST.

En conjunto se busca conseguir un campus más inteligente y mejorar la calidad de vida de sus integrantes dando a conocer al mayor número de personas de manera oportuna sobre los eventos que se realizan periódicamente en la Universidad Industrial de Santander. De esta manera se lograría estimular una cultura universitaria educada en diversos intereses además de los propios y afines de cada persona. Por otra parte, la universidad como tal se vería beneficiada respecto a cómo es vista en el ámbito regional y nacional.

2.2 CONCEPTOS CLAVE EN EL DESARROLLO WEB

2.2.1 Diseño Adaptable. El diseño adaptable es un modo de diseñar y desarrollar sitios web que como su nombre lo indica, su finalidad es que se *adapte* a diferentes tamaños de sitios web. El uso de un diseño adaptable permite una correcta visualización de un sitio web en casi cualquier navegador de cualquier dispositivo que se esté usando, sea móvil o computador de escritorio. Otra ventaja es la de obtener un código eficiente el cual se construye para diferentes plataformas.

Figura 6. Diferentes dispositivos para acceder a Internet

Fuente: ¿QUÉ ES EL DISEÑO WEB ADAPTABLE PARA MÓVILES? [En línea]. (Recuperado en 30 Julio 2019). Disponible en https://usedsources.page.link/DW7Y

Hay al menos dos enfoques por los cuales hacer un desarrollo para un sitio web adaptable:1) *Degradación elegante*, donde partiendo de una página web creada se procede a adaptarla a diferentes tamaños, 2) *Mobile first*, se podría decir que es lo contrario, primero el desarrollo está enfocado a determinado tamaño dispositivo móvil para luego adaptarlo a pantallas más grandes.

A continuación, se mencionan algunos dispositivos donde hoy en día se puede visualizar una página web.

- **Computadores**

En este apartado existe variedad de tamaños de pantalla, las más pequeñas son de los notebooks de 11 pulgadas que aún se usan, y otros más grandes de 28 pulgadas, entre otros.

- **Smartphones**

En la actualidad el uso de smartphone está mundialmente extendido bajo el uso de iOS y Android. Los tamaños de los smartphones más comunes van desde las 4 pulgadas hasta las 6 pulgadas.

- **Tablets**

Suelen tener un tamaño mayor al de los smartphones, los más comunes son de 7 y 8 pulgadas. Otros no tan comunes son los de 10 y 11 pulgadas.

- **TV**

Los smart-tv cuentan con un tamaño de pantalla aún mayor generalmente de más de 32 pulgadas.

2.2.1.1 Algunos Frameworks CSS

- **Foundation:** Foundation es un framework de interfaz de usuario responsive. Foundation proporciona una cuadrícula responsive e incluye componentes de interfaz de usuario HTML y CSS, plantillas, y fragmentos de código, incluyendo tipografía, formularios, botones, barras de navegación y otros componentes de interfaz usuario, así como extensiones de Javascript opcionales. Foundation está mantenida por zurb.com y es un proyecto de código abierto[4].
- **Bootstrap:** Bootstrap es un conjunto de herramientas de código abierto para desarrollar con HTML, CSS y JS. Permite realizar rápidamente un prototipo de ideas o construir aplicaciones completas, sistema de cuadrícula sensible, extensos componentes precompilados y potentes complementos creados en jQuery[5].

[4] GETTING STARTED WITH FOUNDATION 5, What is Foundation?. [En línea]. (Recuperado en 30 de Julio 2019). Disponible en https://foundation.zurb.com/sites/docs/v/5.5.3/
[5] BOOTSTRAP. [En línea]. (Recuperado en 30 de Julio 2019). Disponible en https://getbootstrap.com

- **Bulma:** Bulma se define a sí mismo como: Bulma es un marco CSS gratuito de código abierto basado en Flexbox y utilizado por más de 150.000 desarrolladores[6].
- **Materialize:** Materialize es un framework CSS desarrollado y mantenido por Google por lo que al utilizarlo se le parece mucho a las aplicaciones modernas de Android. Incluye diferentes componentes basado en CSS y otros más basados en JavaScript para, por ejemplo, calendarios y fechas[7].
- **UIKIT:** Este poco conocido framework tiene el mismo enfoque de los anteriores, proveyendo diferentes componentes para un desarrollo web fluido y estéticamente deseable. Tiene una particularidad, y es que sus clases CSS tienen el prefijo **uk-** lo cual permite usarlo junto con otro framework o incluirlo en un desarrollo ya hecho, dado que no va a haber una interferencia y conflicto entre estilos CSS.

2.2.2 Frameworks Javascript MVC. Cuando se habla de *framework,* se refiere a herramientas que facilitan el desarrollo de software. Son marcos de trabajo con un conjunto de conceptos y prácticas previamente definidas enfocadas a resolver un problema específico. Por decirlo de manera sencilla, cada framework define su manera de hacer las cosas.

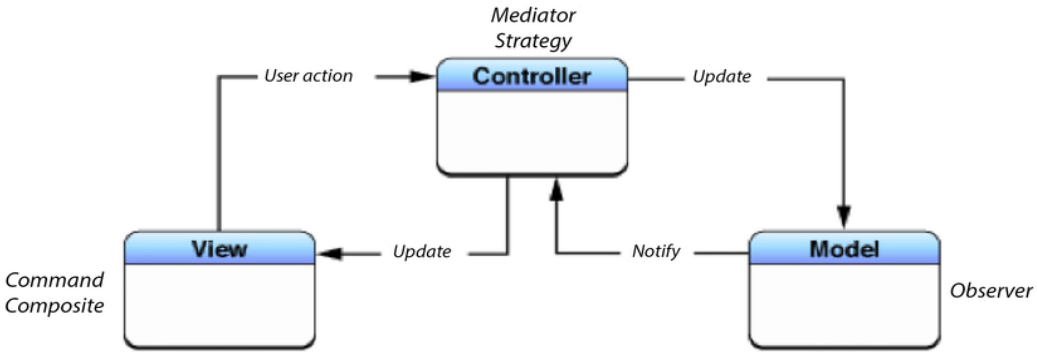
En este caso, es de especial interés aquellos frameworks que se han desarrollado basados en JavaScript (lenguaje de scripting que se ejecuta en el navegador web), ya que debido al avance que ha tenido el desarrollo web, esto ha hecho que estos tipos de aplicaciones cada vez sean más complejas, no solo a nivel visual sino también en la lógica que hay detrás de ello.

Es para manejar dicha *lógica* donde entran en juego estas herramientas pues facilitan el desarrollo comparado a si se hiciera con JavaScript puro. Un enfoque

[6] BULMA. [En línea]. (Recuperado en 30 de Julio 2019). Disponible en https://bulma.io
[7] MATERIALIZE. [En línea]. (Recuperado en 30 de Julio 2019). Disponible en https://materializecss.com

que ha surgido en los frameworks de JavaScript es el conocido **Modelo-Vista-Controlador (MVC).** El objetivo con este tipo de enfoque es tener tres entidades, cada una con su función específica.

Figura 7. Funcionamiento de arquitectura Modelo Vista Controlador



Fuente: Víquez-Acuña, Óscar & Alonso Vega-Brenes, Luis. (2014). Objective C: Análisis de los métodos de comunicación de eventos entre objetos. Revista Tecnología en Marcha. 27. 5. 10.18845/tm. v27i8.2225. (Recuperado en 30 de Julio 2019). Disponible en https://usedsources.page.link/m6ab

El **Modelo** contiene la lógica de los objetos con los que interactúa la aplicación, por ejemplo, usuarios, es el uso más común. Allí también se especifica cómo será su interacción con el servidor.

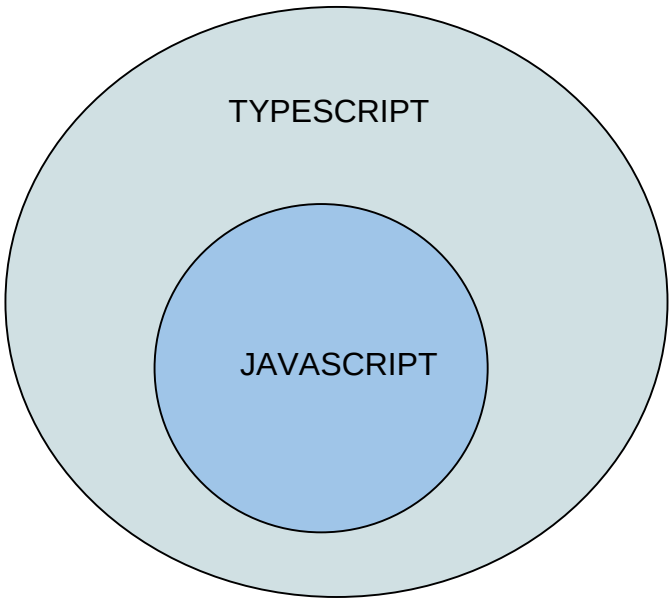
La **Vista,** como se puede intuir, está relacionada con lo que se percibe visualmente. Efectivamente es así, y tiene interacción directa con el **Controlador**, este último se podría decir que es el mediador entre las dos anteriores. El controlador interactúa con el modelo y la vista, llega a conocer los cambios del modelo para que la vista haga los cambios pertinentes. Y viceversa, captura los cambios hechos en la vista y actualiza el modelo.

A continuación, se mencionan solo algunos de los frameworks basados en JavaScript con este enfoque, aunque con ligeras diferencias.

2.2.2.1 Angular. Angular es un framework creado y mantenido por Google, su primera versión se le conocía como AngularJS. Pero luego se reestructuro su código de manera que pasó a llamarse Angular 2. Desde entonces las versiones han aumentado en número, y la acual es Angular 7.

Angular se distingue por el uso de TypeScript, un lenguaje semejante a JavaScript. Que extiende su sintaxis y añade por ejemplo, tipado estático, objetos basados en clases, entre otras cosas[8]. La siguiente figura ilustra la relación entre JavaScript y TypeScript.

Figura 8. Relación entre TypeScript y JavaScript



2.2.2.2 VUE.JS. Vue.js es un marco progresivo para construir interfaces de usuario. A diferencia de otros marcos monolíticos, Vue está diseñado desde cero para ser gradualmente adaptable. La biblioteca central está enfocada solo en la capa de vista, y es fácil de integrar con otras bibliotecas o proyectos existentes. Por otro lado, Vue también es perfectamente capaz de impulsar aplicaciones sofisticadas de

[8] TYPESCRIPT: JAVASCRIPT DEVELOPMENT AT APPLICATION SCALE. [En línea]. (Recuperado en 30 de Julio 2019). Disponible en https://usedsources.page.link/uz1T

una sola página cuando se usa en combinación con herramientas modernas y bibliotecas como soporte.[9]

2.2.2.3 BACKBONE.JS. Backbone provee Modelos, Colecciones, Vistas y Routers que facilitan el desarrollo además de ser completamente compatible con APIs RESTful.

Los **modelos** contienen la información pertinente de los objetos que maneja la aplicación, contiene métodos, atributos, y de ser posible una URL que permite la interacción con el servidor.

Las **colecciones** son conjuntos de modelos, su funcionamiento es semejante, con la ligera diferencia que provee métodos para el fácil manejo de dicho conjunto de modelos.

Las **vistas** contienen la lógica del aspecto visual, generalmente se crea un objeto vista para cada *pantalla* de la aplicación.

Los **routers** permiten conocer la URL a la que el usuario desea navegar, dependiendo de esos cambios el router llama a la vista correspondiente.

2.2.3 Visualización de Gráficos en la Web. En la mayoría de los casos, en el desarrollo web es necesario mostrar cierta cantidad de datos de una manera visualmente entendible y por ello han surgido diferentes librerías que se encargan de hacer este trabajo y lo hacen usando JavaScript.

[9] WHAT IS VUE.JS?. [En línea]. (Recuperado en 25 Julio 2019) Disponible en https://vuejs.org/v2/guide/

Las tablas son el tipo de gráfico más simple, y durante años fue común el uso de tablas HTML, no obstante, para el caso de grandes cantidades de datos, no siempre son útiles las tablas, y no siempre es posible crear tabla por tabla.

Existen numerosas librerías que se encargan de este apartado, a continuación, se mencionan algunas.

2.2.3.1 CHARTIST.JS. Chartist.js es una librería ligera que solo pesa 10 Kb comprimida. Es bastante sencilla de usar y sus gráficos se renderizan usando SVG, lo cual lo hace compatible con la mayoría de los navegadores actuales. También permite personalizar sus estilos con CSS y es adaptable a diferentes tamaños de pantalla.

2.2.3.2 CHARTS.JS. Charts.js también permite personalización a través de estilos CSS dedicados. Cuenta con animaciones que permiten una óptima experiencia de usuario y también es posible que sus gráficos se adapten al tamaño de la pantalla.

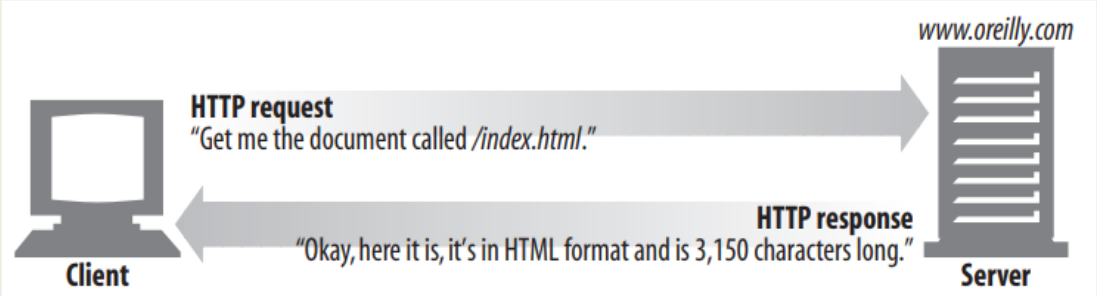
2.2.3.3 D3.JS. D3.js es peculiarmente diferente de otras librerías porque su enfoque va más allá. Ha sido creado con la capacidad para mostrar gráficos complejos que son útiles, por ejemplo, en otros campos como la física y química. D3.js también renderiza sus gráficos usando SVG y está enfocado a grandes datasets.

2.2.3.4 Google Charts. Google también cuenta con su librería de gráficos, esta al igual que las anteriores, dibuja sus gráficos en formato SVG y VML (para versiones antiguas de Internet Explorer). En su galería de gráficos cuenta con alrededor 28 diferentes tipos de gráficos. Tiene varias opciones de personalización y también para la obtención de los datos, permitiendo, por ejemplo, obtenerlos de una hoja de cálculo de Google.

2.2.4 API Rest. En este apartado se analiza la comunicación de una aplicación web con el servidor. El servidor es la máquina que sirve la información que necesita la aplicación. La comunicación con el servidor se realiza para obtener información, por lo tanto, durante el uso de la aplicación hay un intercambio en ambos sentidos de dicha información. Dicha comunicación ocurre usando un *idioma* conocido por ambas partes para entenderse.

2.2.4.1 HTTP. En la web se usa ampliamente el conocido protocolo HTTP, el cual permite hacer solicitudes y enviar respuestas. Está diseñado para que cada objeto, solicitud o respuesta, contenga la información necesaria para que su receptor entienda lo que se le pide o lo que necesitaba conocer.

Figura 9. Interacción entre cliente y servidor usando HTTP

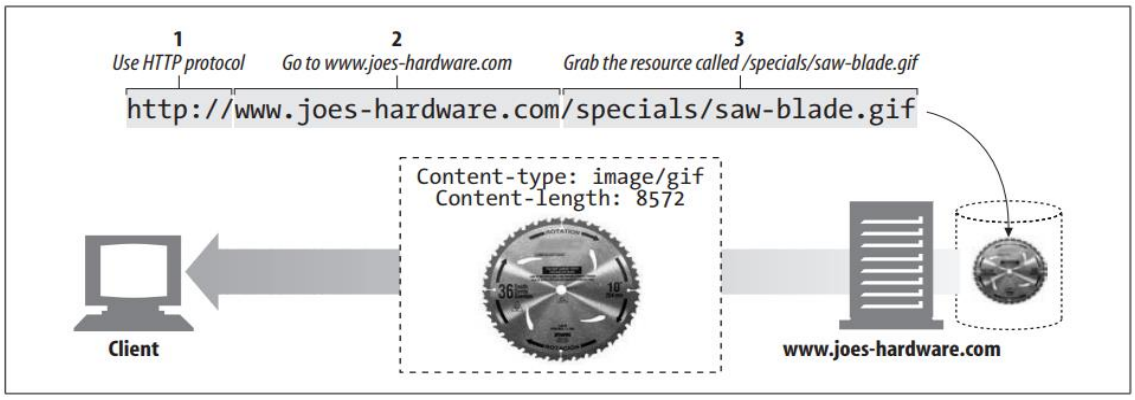


Fuente: GOURLEY David, TOTTY Brian. HTTP: The Definitive Guide. O'Reilly Media, 2002 p.43. ISBN-10: 1-56592-509-2 ISBN-13: 978-1-56592-509-0

Como se puede ver en la figura, todo funciona a base de solicitudes y respuestas al servidor. Cuando se hace una solicitud, se intenta acceder a un recurso. Un recurso es cualquier fuente de contenido, imágenes, archivos de texto, archivos comprimidos, etc…

Al intentar acceder a un recurso, se suele usar una URL (Uniform Resource Locator). El servidor recibe la petición y busca el recurso en dicha URL para luego enviarlo al cliente.

Figura 10. Solicitud HTTP a un recurso



Fuente: GOURLEY David, TOTTY Brian. HTTP: The Definitive Guide. O'Reilly Media, 2002 p.43. ISBN-10: 1-56592-509-2 ISBN-13: 978-1-56592-509-0

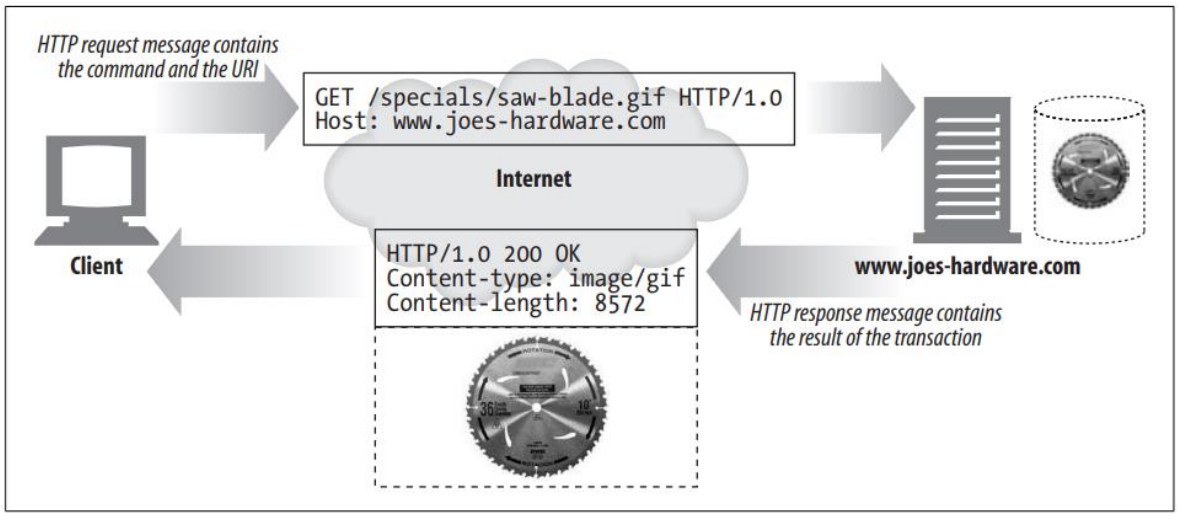
2.2.4.2 JSON. JavaScript Object Notation, se deriva de JavaScript, es un formato de intercambio muy popular [JavaScript and JSON Essentials] en el cual se encapsula la información y es de hecho más fácil de entender, en contraposición al formato XML, que puede ser más confuso. El uso de JSON se ha extendido, numerosos lenguajes tienen soporte para este formato: JavaScript, Python, PHP, Java, C++, Ruby, entre otros. Al contar con librerías o soporte de forma nativa hace más fácil su uso.

2.2.4.3 REST. Teniendo en cuenta cómo funciona HTTP y que es utilizado para la interacción entre cliente y servidor, ahora veremos que significa REST.

REST significa Transferencia de Estado Representacional. "REST es un estilo de arquitectura software para sistemas hipermedia distribuidos como la World Wide Web.

La arquitectura REST aprovecha la existencia de las URIs (Uniform Resource Identifier) para obtener los recursos. Una URI identifica de manera única a un recurso. Entonces, el enfoque es básicamente, siempre hacer peticiones a recursos.

Figura 11. Petición a una URI



Fuente: GOURLEY David, TOTTY Brian. HTTP: The Definitive Guide. O'Reilly Media, 2002 p.43. ISBN-10: 1-56592-509-2 ISBN-13: 978-1-56592-509-0

Como se mencionó antes, REST es una arquitectura, por lo tanto, plantea una serie de pautas para comunicarse con el servidor y también para estructurar las URI de los distintos recursos. Deben seguirse entonces estos lineamientos, cuando esto se logra, se dice que hemos diseñado una *API REST*.

La razón por la que se le llama API es porque según se define, API significa Interfaz de Programación de Aplicaciones. Como interfaz, provee una capa de abstracción entre dos sujetos que interactúan, una API facilita la interacción de los dos permitiendo la claridad en la comunicación.

Por consiguiente, una API REST es una serie de funciones o procedimientos a través de los cuales se comunican cliente y servidor, que, como se dijo, funciona como una capa de abstracción para que el cliente no sepa cómo están ubicados realmente de manera local los recursos en el servidor.

A continuación, se resumen las pautas para diseñar una API REST.

- Las API de REST se diseñan en torno a recursos, que son cualquier tipo de objeto, dato o servicio al que puede acceder el cliente. Por ejemplo: https://adventure-works.com/orders/1
- El formato de intercambio de información puede ser XML o JSON (JavaScript Object Notation), aunque lo más común es usar JSON.
- Al estar diseñada sobre HTTP, hace uso de los verbos disponibles, GET, POST, PUT, PATCH y DELETE.

En la siguiente figura se muestra cómo puede funcionar las solicitudes dependiendo del verbo que se use.

Figura 12. Solicitudes entorno a recursos

Recurso	POST	GET	PUT	DELETE
/customers	Crear un nuevo cliente	Recuperar todos los clientes	Actualización masiva de clientes	Eliminar todos los clientes
/customers/1	Error	Recuperar los detalles del cliente 1	Actualizar los detalles del cliente 1 si existe	Quitar al cliente 1
/customers/1/orders	Crear un nuevo pedido para el cliente 1	Recuperar todos los pedidos del cliente 1	Actualización masiva de pedidos del cliente 1	Recuperar todos los pedidos del cliente 1

Fuente: DISEÑO DE API. [En línea]. (Recuperado en 30 de Julio 2019). Disponible en https://usedsources.page.link/fFWD

En vista de lo anterior, una arquitectura REST permite la escalabilidad, de ser necesario implementar el acceso a más recursos. El uso de verbos es intuitivo según el recurso al cual se accede y dado que lo más común es usar JSON como formato de intercambio, esto hace que sea más fácil su implementación, no solo con un cliente web sino un cliente móvil (o smartphone).

2.4 DEFINICIÓN DE USABILIDAD

La usabilidad tiene varios componentes y se le asocian cinco componentes:

- **Capacidad de aprendizaje:** El sistema es fácil de aprender a usar. El usuario puede rápidamente hacer una tarea.
- **Eficiencia:** Una vez aprendido de usar, el usuario tiene alta productividad usando el sistema
- **Fácil de recordar:** El usuario puede recordar fácilmente como usar el sistema después de largo tiempo sin usarlo.
- **Manejo de errores:** El sistema tiene una baja tasa de errores. En el caso de que falle, el usuario puede volver a un punto antes del error.
- **Satisfacción:** El sistema es agradable de usar

Cuando un sistema cumple con estos atributos tiene una alta medida de usabilidad para sus usuarios, aunque claro está, todo esto depende del usuario y la tarea que desea realizar.

2.5 ESTADO DEL ARTE

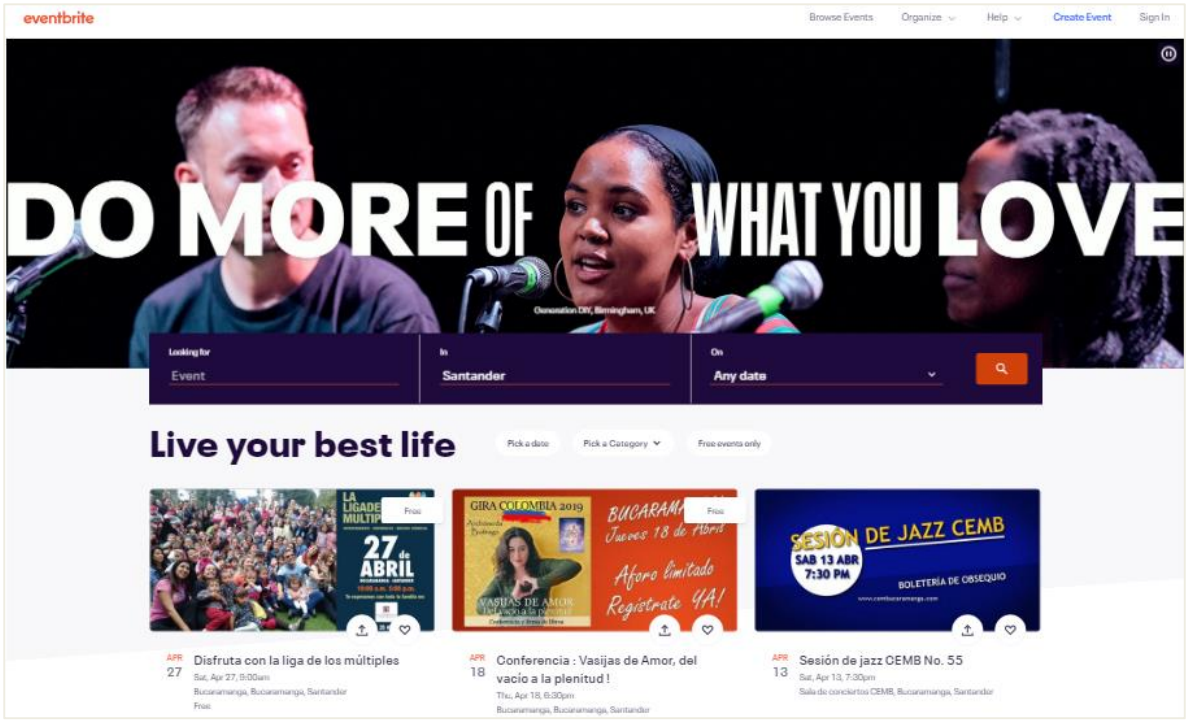
La realización de eventos y su seguimiento es una actividad muy común, constantemente deseamos saber cuándo sucede algo de nuestro interés. A este respecto existe por ejemplo una plataforma llamada *EventBrite.*

Eventbrite es una plataforma global para experiencias en vivo que permite a cualquier persona crear, compartir, encontrar y asistir a eventos que estimulen sus pasiones y enriquezcan sus vidas. Desde festivales de música, maratones, conferencias, reuniones comunitarias y eventos para recaudar fondos, hasta

concursos de juegos y concursos de guitarra de aire. Nuestra misión es unir al mundo a través de experiencias en vivo[10].

Como ellos mismos definen, es una plataforma a nivel global, es decir, funciona en Colombia y otros países.

Figura 13. Página principal EventBrite



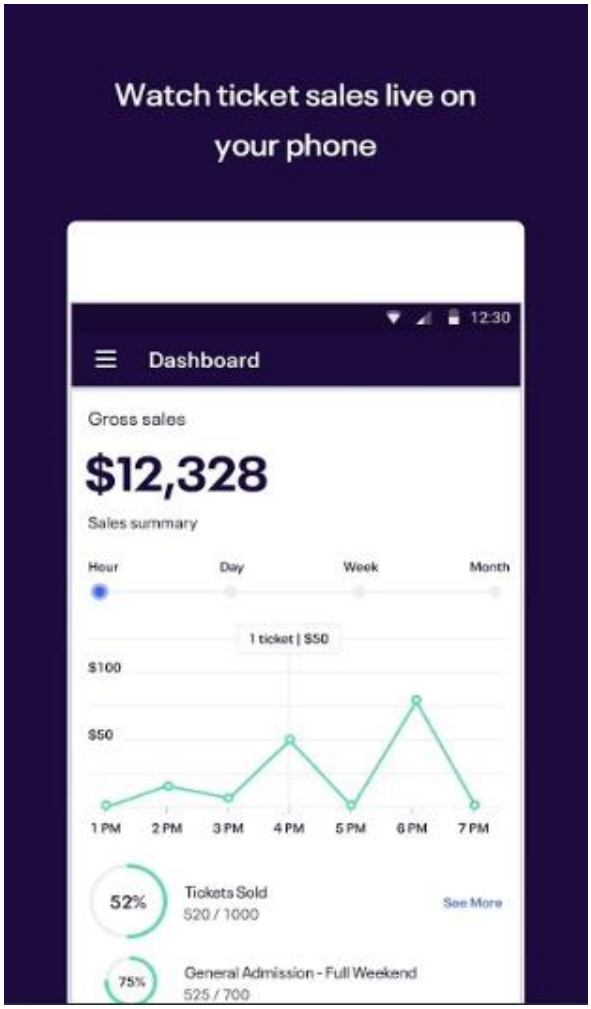
Fuente: EVENTBRITE HOME PAGE. [En línea]. (Recuperado en 25 Julio 2019) Disponible en https://www.eventbrite.com

Permite organizar y crear eventos de diferentes temáticas, además de buscarlos haciendo uso de diferentes filtros.

[10] EVENTBRITE - ABOUT US. [En línea]. (Recuperado en 25 Julio 2019) Disponible en https://www.eventbrite.com/about/

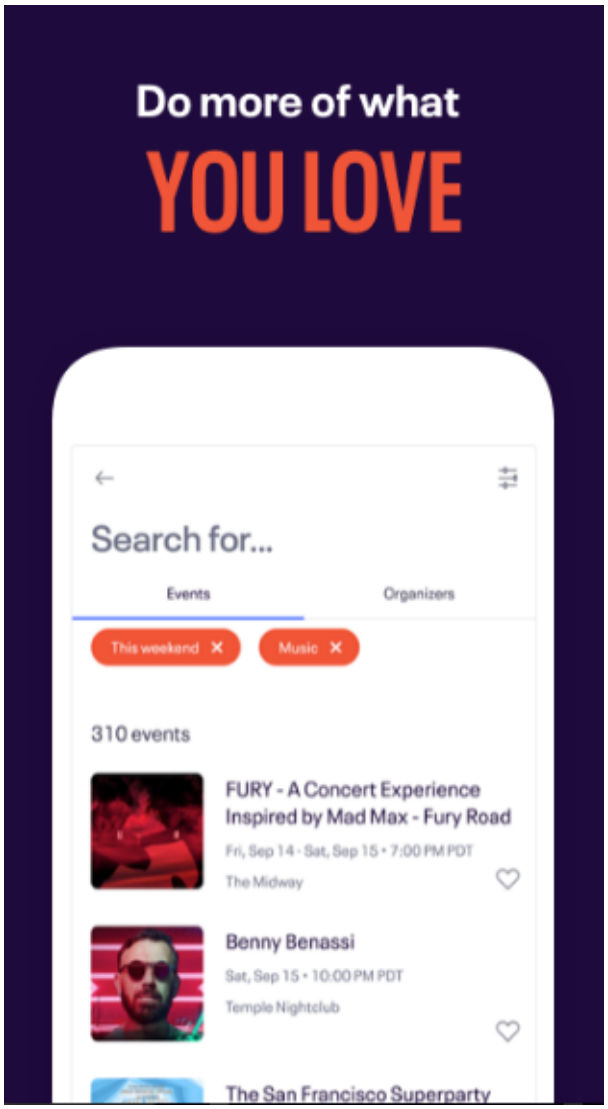
EventBrite cuenta también con dos aplicaciones móviles desarrolladas para iOS y Android. Una de ellas se enfoca en el usuario organizador de los eventos, y la otra en el usuario que asiste a los eventos.

Figura 14. Aplicación móvil: Eventbrite organizador



Fuente: GOOGLE PLAY - EVENTBRITE ORGANIZADOR. [En línea]. (Recuperado en 30 de Julio 2019). Disponible en https://play.google.com/store/apps/details?id=com.eventbrite.organizer

Figura 15. EventBrite (para visualizar eventos)



Fuente: EVENTBRITE - DESCUBRE EVENTOS Y DIVERSIÓN CERCA. [En línea]. (Recuperado en 30 de Julio 2019). Disponible en https://usedsources.page.link/KfNz

EventBrite es una plataforma completa tanto en funcionalidad como en disposición para usarlo desde diferentes dispositivos. Es por ello que el desarrollo del presente proyecto considero algunas características de EventBrite, básicamente se ha usado como guía.

3. METODOLOGÍA

Para este proyecto se hará uso de una adaptación de prototipos evolutivos. Esta elección se ha hecho teniendo en cuenta que dicho modelo permite prototipos usables en poco tiempo, pudiendo así tener una retroalimentación en cada iteración y hacer una refinación que permita acercarse al producto final deseado.

3.1 CONTEXTUALIZACIÓN DEL DOMINIO Y TECNOLÓGICA

Actividad. En esta fase se pretende familiarizarse con los detalles sobre la organización de un evento y su contextualización de aspectos técnicos.
Consta de una investigación enfocada a estos detalles
Producto. Conceptos implicados claros y documento de requerimientos

3.2 IDENTIFICACIÓN DE NECESIDADES Y ESPECIFICACIÓN DE LA ARQUITECTURA

Actividad. Se hará un estudio y posterior selección de la arquitectura de software que mejor se adapte a las necesidades de una plataforma de este tipo.

Es importante recalcar que en esta fase será necesario llevar a cabo reuniones con quienes en primera instancia serán los usuarios finales.
Producto. Propuesta de la arquitectura a utilizar

3.3 PROTOTIPADO

Actividad. El enfoque será el desarrollo como tal de la plataforma cuyo principal esfuerzo estará en proveer al administrador de eventos una interfaz sencilla,

completa y fácil de usar al momento de publicar un evento. Se incluye también la funcionalidad de obtener estadísticas acerca de un evento determinado.
Producto. Producto terminado.

3.4 PRUEBAS

Actividad. Se llevará a cabo pruebas funcionales haciendo uso del instrumento encuesta para determinar la aceptación de la aplicación por parte de los usuarios.
Producto. Producto probado correctamente

3.5 PROTOTIPO FINAL

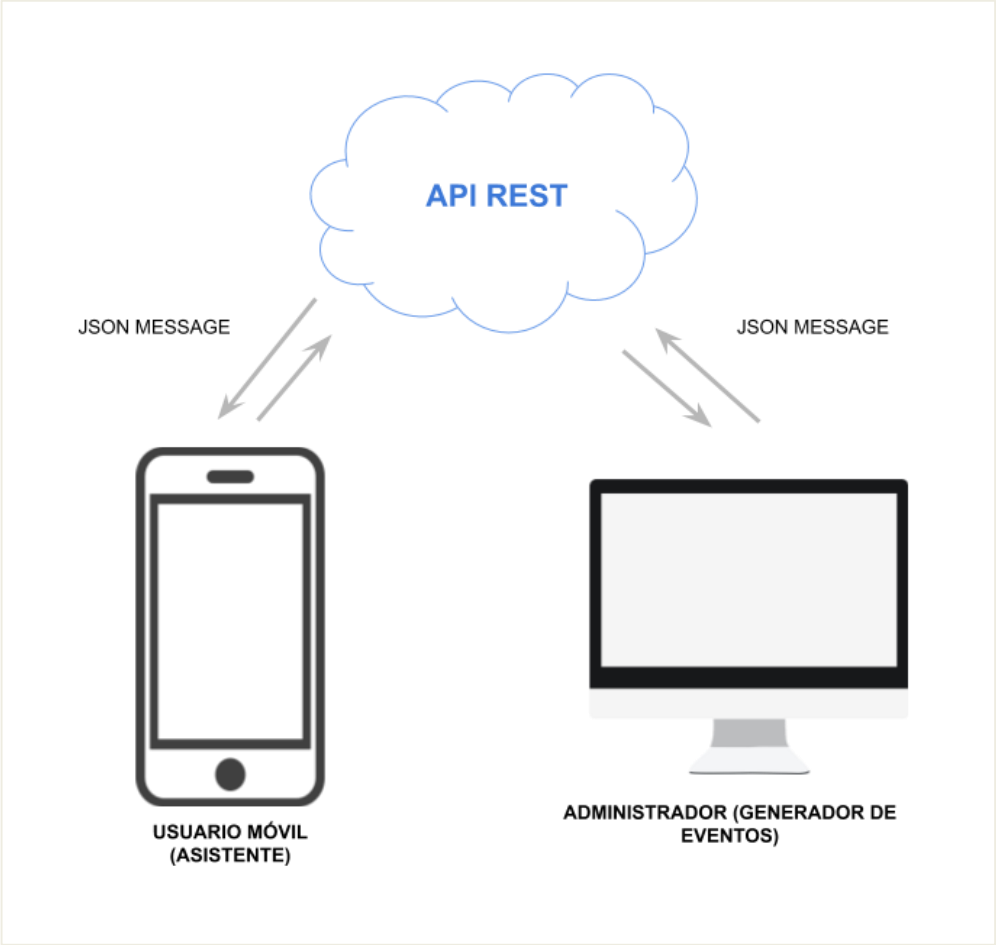
Actividad. Pruebas finales del prototipo
Producto. Prototipo final terminado.

4. DISEÑO E IMPLEMENTACIÓN

4.1 ARQUITECTURA GLOBAL

Este proyecto hace parte de un sistema el cual tiene un componente back-end y móvil (Android). Por ende, se ha planteado la siguiente arquitectura global del sistema:

Figura 16. Arquitectura global del sistema



En la figura anterior vemos que, cada tipo de usuario, móvil y administrador, se comunica con el backend mediante un API REST usando llamadas HTTP en formato JSON (JavaScript Object Notation).

El usuario asistente consulta la información disponible única y exclusivamente desde su teléfono móvil usando la aplicación desarrollada en Android para este macroproyecto.

El usuario administrador de eventos publica eventos y obtiene sus los respectivos datos de la interacción con los asistentes mediante la aplicación web desarrollada en el presente proyecto.

4.2 DISEÑO DE LA SOLUCIÓN

4.2.1 Identificación de Necesidades

Tabla 1. Necesidades a suplir por la aplicación

Necesidad	Descripción
Publicar eventos de diferentes categorías	Crear y publicar eventos de diferentes categorías
Cambiar de estado a los eventos	Cambiar estado del evento a voluntad del usuario
Filtrar eventos por estado	Obtener eventos de acuerdo al estado que este tenga
Visualizar los detalles de un evento	Permite ver detalles de un evento
Modificar un evento	Modificar la información asociada a un evento
Crear un evento	Crear un evento cuyo estado es creado
Subir imágenes de un evento creado	Si el usuario lo decide, puede cargar imágenes cuando crea un evento
Modificar imágenes de un evento existente	Si el usuario lo decide, puede cargar nuevas imágenes cuando modifica un evento

Necesidad	Descripción
Mostrar gráficos sobre los atributos de un evento	Permite ver los atributos de un evento cuando este se ha realizado
Filtrar eventos de pago	Permite filtrar los eventos si son de pagos o gratis
Filtrar eventos entre dos fechas	Permite filtrar eventos entre dos fechas, la fecha actual y otra seleccionada por el usuario
Diseño adaptable	La aplicación tiene un diseño adaptable a dispositivos móviles

4.2.2 Requerimientos del Sistema. A continuación, se muestran los requerimientos funcionales y no funcionales de la aplicación.

Tabla 2. Requerimientos funcionales

REQUISITO	DESCRIPCION
RE001	La aplicación debe permitir registrar usuario, para ello requerirá correo, nombre, apellidos, organización, contraseña, nombre de usuario.
RE002	La aplicación permitirá iniciar sesión al usuario usando nombre de usuario y contraseña.
RE003	La aplicación permitirá modificar información del usuario, únicamente nombres, apellidos, organización, email.
RE004	La aplicación permitirá al usuario cambiar su contraseña siempre y cuando especifique su contraseña actual.
RE005	La aplicación mostrará los eventos registrados del usuario que ha iniciado sesión organizándolos en gráficos y como tarjetas.
RE006	El usuario podrá crear un evento diligenciando la información necesaria: nombre, ciudad, lugar, fecha, hora, categoría, capacidad. El evento no podrá tener una fecha anterior al día en que se esté creando el evento.
RE007	Si el usuario desea cargar imágenes a un evento nuevo, el usuario podrá cargarlas después de diligenciar la información necesaria para crear el evento. Las imágenes no
RE008	El usuario podrá modificar un evento y guardar los cambios en el servidor. La fecha no podrá cambiarse a una anterior al día en que se esté modificando el evento.

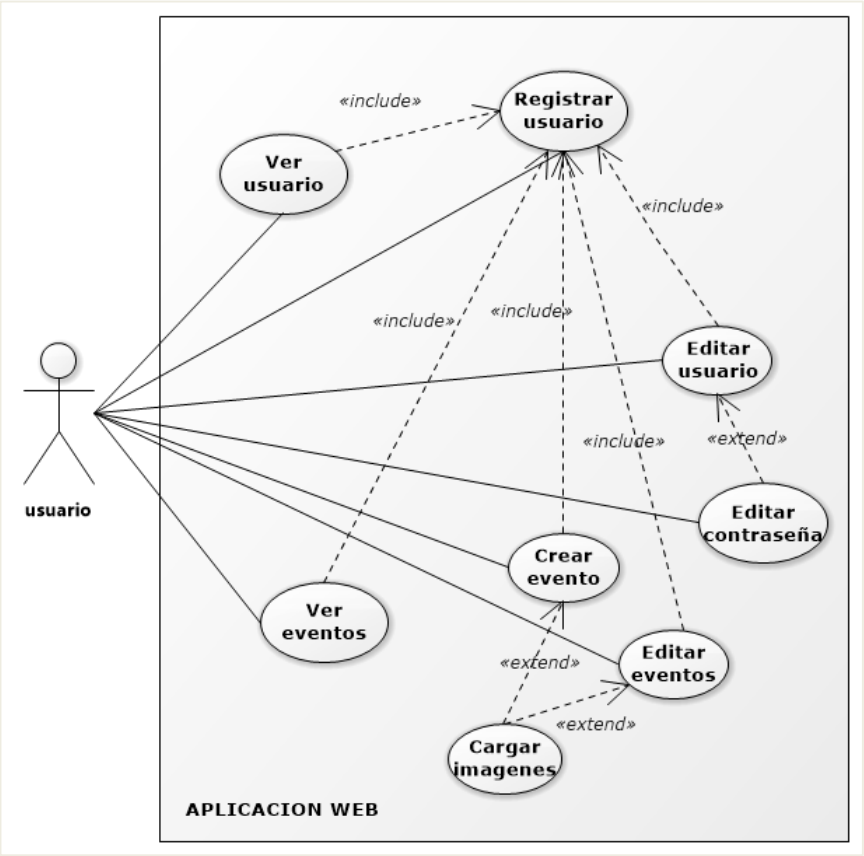
Ahora se mostrarán los requerimientos no funcionales de la aplicación.

Tabla 3. Requerimientos no funcionales

REQUERIMIENTO	DESCRIPCION
RE009	La aplicación se adaptara a tamaños de pantalla inferiores a 768px, que corresponden a tablets y smartphones.
RE010	Los usuarios podrán manejar fácilmente la aplicación mediante accesos rápidos a lo largo de la navegación.

4.2.3 Casos de Uso. A continuación, se muestra el diagrama de casos de uso.

Figura 17. Diagramas de caso de uso



En el diagrama de casos de uso podemos ver las diferentes acciones que realiza cada actor.

El usuario puede registrarse, ver y editar su información, crear eventos y cargar imágenes. El servidor por su parte también está involucrado en casi todas estas actividades, pero solo cuando recibe la información que el usuario edita y desea guardar.

4.2.4 Especificación de Casos de Uso. A continuación, se muestran la especificación de los casos de uso.

Tabla 4. Caso de uso: Registrar usuario

ID de Caso de uso	01
Nombre del caso de uso	Registrar usuario
Características	Registrar usuario en la aplicación
Descripción del caso de uso	El usuario administrador de los eventos digita la información necesaria en el formulario de registro para luego iniciar sesión en la aplicación.
Condiciones de entrada	El usuario no está registrado en el sistema
Condiciones de salida	El usuario está registrado en el sistema
Instancias de actores participantes	Usuario

Tabla 5. Caso de uso: Iniciar sesión

ID de Caso de uso	02
Nombre del caso de uso	Iniciar sesión
Características	Iniciar sesión en la aplicación
Descripción del caso de uso	El usuario administrador de los eventos digita la información necesaria para iniciar sesión.
Condiciones de entrada	El usuario está registrado en el sistema
Condiciones de salida	El usuario inicia sesión en la aplicacion
Instancias de actores participantes	Usuario

Tabla 6. Editar información de usuario

ID de Caso de uso	03
Nombre del caso de uso	Editar información de usuario
Características	Editar información como correo, nombre, apellidos.
Descripción del caso de uso	El usuario administrador de los eventos puede editar su propia información a excepción de la contraseña.
Condiciones de entrada	El usuario está registrado en el sistema

Condiciones de salida	El usuario modifica su información de usuario excepto su contraseña
Instancias de actores participantes	Usuario

Tabla 7. Editar contraseña de usuario

ID de Caso de uso	**04**
Nombre del caso de uso	Editar contraseña usuario
Características	Editar contraseña de usuario en la aplicación
Descripción del caso de uso	El usuario administrador de los eventos puede editar su contraseña de ingreso a la aplicación validando mediante proporcionar su contraseña antigua.
Condiciones de entrada	El usuario está registrado en el sistema
Condiciones de salida	El usuario modifica su contraseña de ingreso a la aplicación.
Instancias de actores participantes	Usuario

Tabla 8. Caso de uso: Crear evento

ID de Caso de uso	**05**
Nombre del caso de uso	Crear evento
Características	Crear evento en la aplicación
Descripción del caso de uso	El usuario administrador de los eventos digita la información requerida para crear un evento. Posteriormente esos cambios son enviados al servidor.
Condiciones de entrada	El usuario está registrado en el sistema
Condiciones de salida	El usuario crea un evento en el sistema
Instancias de actores participantes	Usuario

Tabla 9. Caso de uso: Ver evento

ID de Caso de uso	06
Nombre del caso de uso	Ver evento
Características	Ver eventos registrados
Descripción del caso de uso	El usuario administrador de eventos puede ver la información que contiene cada evento que tenga registrado
Condiciones de entrada	El usuario está registrado en el sistema
Condiciones de salida	El usuario ve los eventos que tiene registrados en el sistema
Instancias de actores participantes	Usuario

Tabla 10. Caso de uso: Editar evento

ID de Caso de uso	07
Nombre del caso de uso	Editar evento
Características	Editar eventos registrados
Descripción del caso de uso	El usuario administrador de los eventos puede editar cualquier evento que tenga registrado.
Condiciones de entrada	El usuario está registrado en el sistema
Condiciones de salida	El usuario edita un evento que haya seleccionado.
Instancias de actores participantes	Usuario

Tabla 11. Caso de uso: Cargar imágenes a un evento

ID de Caso de uso	08
Nombre del caso de uso	Cargar imágenes a un evento
Características	Registrar usuario en la aplicación
Descripción del caso de uso	El usuario administrador de los eventos puede seleccionar imágenes desde su máquina local y cargarlas a un evento que esté editando y guardarlas en el servidor.

Condiciones de entrada	El usuario está registrado en el sistema
Condiciones de salida	El usuario carga imágenes nuevas al evento seleccionado.
Instancias de actores participantes	Usuario

4.2.5 Selección de Herramientas.

Node.js: Node.js es un entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome[11]. La diferencia con lo que comúnmente se conoce de JavaScript es que se ejecuta en el navegador, pero Node.js se utiliza especialmente para desarrollar aplicaciones que se ejecutan del lado del servidor.
Para este proyecto se hizo uso de Node.js aunque el back-end propio de la aplicación estuvo a cargo de otro estudiante de Ingeniería de Sistemas, la razón es que, debido a la dificultad de hacer comprobaciones conjuntas a medida que avanzaba el desarrollo, se vio más pertinente usar un servidor basado en Node.js que proveyera datos simulados como hubiera ocurrido en el otro caso. Otra razón por la que fue escogido es que Node.js permite desarrollar servicios REST más rápidamente que otras alternativas listadas en la tabla 13, además de la familiaridad con JavaScript que posee el autor del proyecto.

Backbone.js: Backbone.js es un framework MVC que permite el desarrollo de aplicaciones web con la capacidad modularizar el código[12] facilitando así su posterior mantenimiento. La curva de aprendizaje de backbone.js es más suave que otros frameworks JavaScript con este mismo enfoque, además al permitir estructurar mejor el código al momento de desarrollar logra que este sea más fácil de mantener o cambiar. Es por esto que fue seleccionado por encima de las otras alternativas que se listan en la tabla 13.

[11] NODEJS. [En línea] (Recuperado en 30 de Julio 2019). Disponible en https://nodejs.org/es/
[12] BACKBONE.JS. [En línea]. (Recuperado en 29 Julio 2019). Disponible en https://backbonejs.org

Tabla 12. Comparativa Backbone.js y otros frameworks

Framework	Ventajas	Desventajas
Backbone.js	● Diseñado para comunicarse con APIs REST. ● Más flexible. ● Más fácil de aprender. ● Posibilidad de organizar mucho mejor el código.	● Más pequeño comparado con los demás, por lo tanto, necesita más complementos. ● Documentación menos detallada
Angular	● Más documentación de parte de google. ● Mejor soporte de la comunidad. ● Fácil de probar.	● Se deben seguir los estándares planteados por Google en el desarrollo. ● Curva de aprendizaje más empinada. ● No está enfocado a comunicarse con API REST.
Vue.js	● Documentación fácil de entender. ● Permite desarrollar aplicaciones de una sola página completamente o solo componentes de aplicaciones existentes.	● Más complicado de aprender que backbone.js, pero más fácil que angular. ● No está enfocado a comunicarse con API REST.

Tabla 13. Comparativa con Node.js

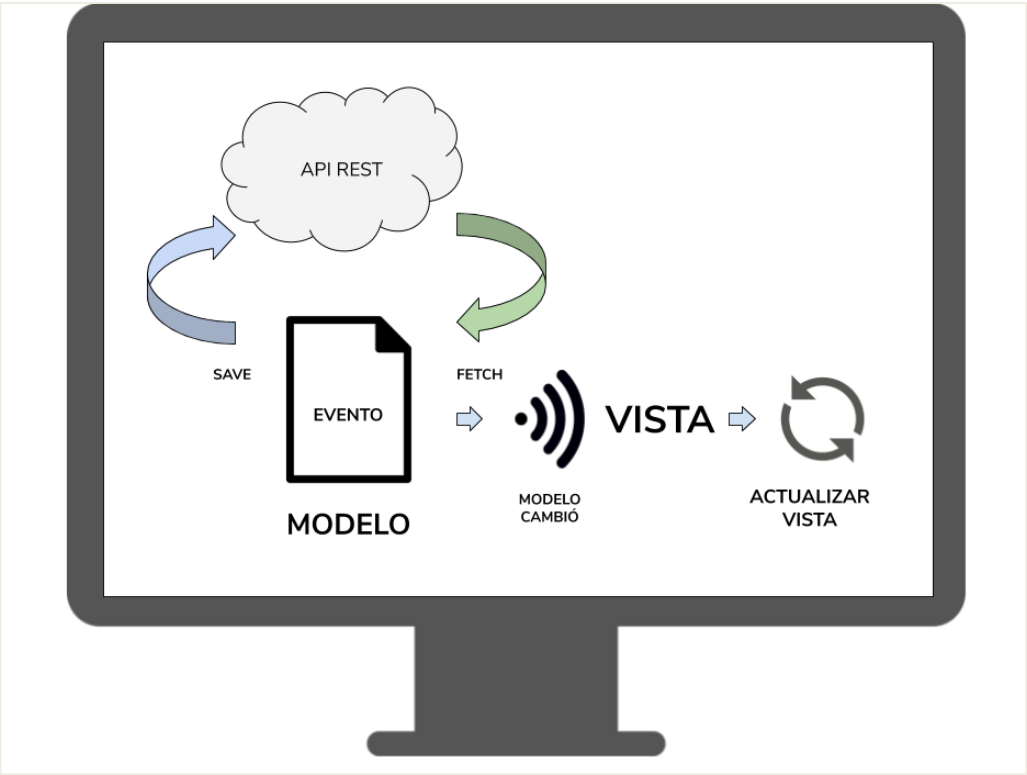
Herramienta	Ventajas	Desventajas
Node.js	● Permite desarrollo de aplicaciones front-end y back-end ● Usa el gestor de paquetes npm ● Es más rápido levantar un servidor basado en Node.js	● Más ineficiente al manejar recursos de hardware.
Spring Boot	● Posee tipado estático, y con ello se evitan errores de escritura de código y fácil depuración. ● Se ejecuta en la JVM, que tiene un uso eficiente de recursos	● Instalar software necesario requiere más tiempo. ● Se basa en un lenguaje diferente al usado en el front-end
Flask	● Permite un desarrollo ágil de servicios backend ● Permite desarrollar API REST	● Se basa en un lenguaje diferente al usado en el front-end.

4.2.6 Arquitectura de la Solución. Al usar Backbone.js como framework principal de JavaScript, este provee la funcionalidad de desarrollo conocido como Modelo-Vista-Controlador. Backbone.js hace uso de los modelos, los cuales almacenan la información de las entidades que interactúan en una aplicación. Backbone.js

también tiene colecciones, estas son básicamente un conjunto de modelos. Las colecciones y los modelos poseen los métodos que interactúan con el servidor.

Teniendo en cuenta lo anterior, la arquitectura propuesta fue la siguiente:

Figura 18. Arquitectura de la aplicación

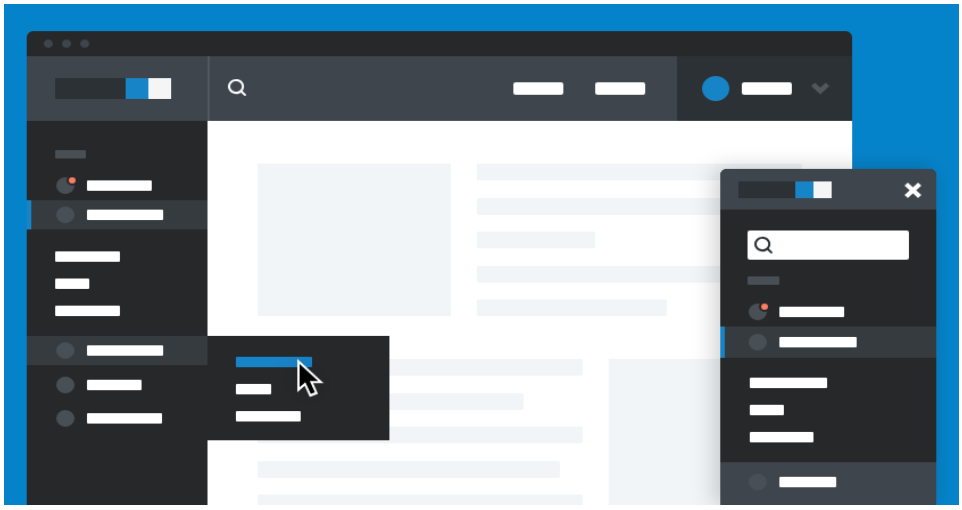


El usuario administrador interactúa con las diferentes vistas de la aplicación. Cada modificación que pueda hacer en la respectiva vista se envía automáticamente al modelo asociado a esa vista, por ende, el modelo cambia, es decir cambia la información que contiene al guardar los cambios enviándolos al servidor usando métodos basados en AJAX, en este caso **save**. Una vez el modelo ejecuta este método, la colección que almacena ese modelo llama al método **fetch**, que se encarga de traer los últimos cambios del servidor. Todos los modelos contenidos en esa colección son *devueltos* por el servidor y el modelo en cuestión queda

actualizado. A continuación, el modelo notifica a la vista que ha habido cambios y esta se refresca para mostrar el ultimo estado del modelo. Esto es un ciclo continuo.

4.2.7 Interfaz de la Solución. La interfaz gráfica de la aplicación no fue un desarrollo propio. Se ha usado un componente o plantilla de navegación tipo dashboard del framework CodyHouse[13] desarrollado por *Amber Creative Lab*[14]. El componente fue diseñado con la metodología **mobile-first,** por ello la interfaz es adaptable a dispositivos móviles.

Figura 19. Plantilla para interfaz gráfica de la solución



Fuente: RESPONSIVE SIDEBAR NAVIGATION. [En línea]. (Recuperado en 29 Julio 2019). Disponible en https://usedsources.page.link/CPo8

4.2.8 Diseño de Datos. Las dos clases que interactúan a lo largo de la ejecución de la aplicación son **Evento** y **Usuario**. A continuación, se muestran los atributos de la clase Evento.

[13] CODYHOUSE - FRAMEWORK. [En línea]. (Recuperado en 29 Julio 2019). Disponible en https://codyhouse.co/ds/docs/framework
[14] AMBER CREATIVE LAB. [En línea]. (Recuperado en 15 Agosto 2019). Disponible en https://ambercreative.co

Tabla 14. Atributos clase Evento

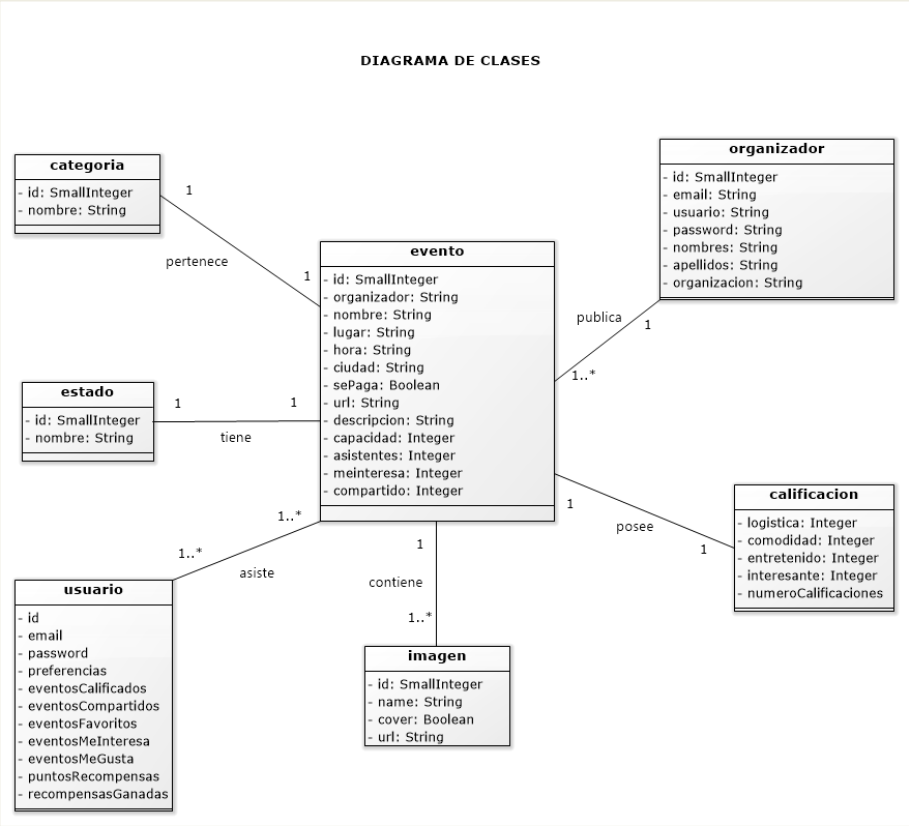
Atributo	Tipo	Ejemplo
id	number	210
organizador	string	“Brayangf”
nombre	string	“Festival Internacional de Piano”
lugar	string	“Auditorio Luis A. Calvo”
hora	string	“19:00”
ciudad	string	“Bucaramanga”
categoria	string	“musica”
sePaga	boolean	false
url	string	“https://www.uis.edu.co/webUIS/es/index.jsp”
descripcion	string	“Evento creado para la comunidad UIS”
capacidad	number	987
asistentes	number	0
meinteresa	number	45
compartido	number	15
estado	string	“publicado”
calificacion	object	{logistica:{},comodidad:{},entretenido:{},interesante:{}}
calificacion.logistica	object	{“1”:4,”2”:9,”3”:,”4”:7,”5”:6}
calificacion.comodidad	object	{“1”:4,”2”:9,”3”:,”4”:7,”5”:6}
calificacion.entretenido	object	{“1”:4,”2”:9,”3”:,”4”:7,”5”:6}
calificacion.interesante	object	{“1”:4,”2”:9,”3”:,”4”:7,”5”:6}
imagenes	array	[{},{},{}]
imagenes[0]	object	{url:””,name:””,cover:true}
imagenes[0].url	string	“https://i.ytimg.com/vi/0zPOSOqTh5c/hqdefault.jpg”
imagenes[0].name	string	“ad794ff333fcb22e31a4c4c”
imagenes[0].cover	boolean	true

Tabla 15. Atributos clase Usuario (organizador)

Atributo	Tipo	Ejemplo
id	number	1
email	string	“brayangf10@yahoo.com”
usuario	string	“Brayangf”
password	string	“000000”
nombres	string	“Brayan”
apellidos	string	“Guerrero”
organizacion	string	“UIS”

4.2.9 Diagrama de Clases. A continuación, se muestra el diagrama de clases de la aplicación.

Figura 20. Diagrama de clases de la aplicación



El anterior diagrama de clases proporciona una visión conceptual de las entidades que existen e interactúan durante la ejecución de la aplicación.

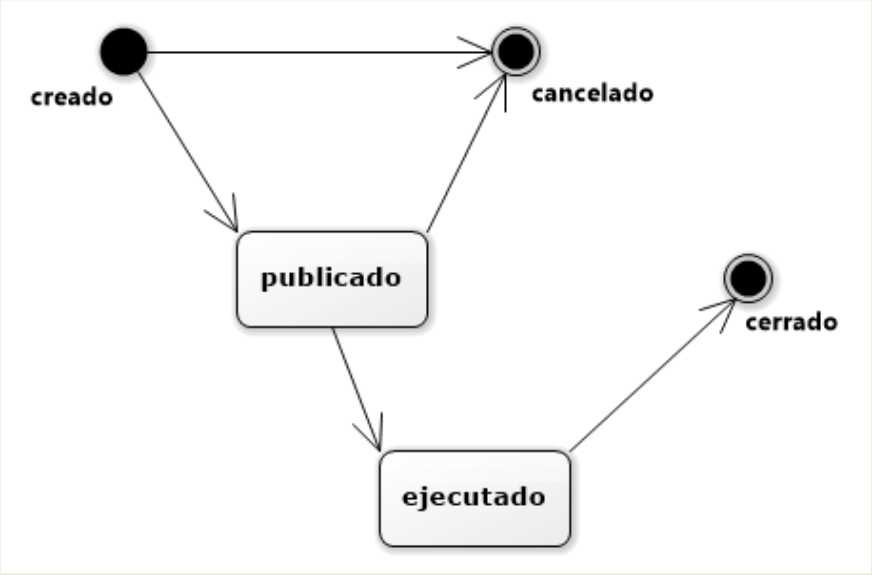
Estas clases son: categoría, estado, usuario, imagen, evento, organizador, calificación.

Vemos que el usuario puede asistir a uno o más eventos, sus atributos muestran la interacción con los eventos que ha creado el organizador. El organizador puede ver estos cambios realizados por el asistente.

Un evento posee una calificación que se obtiene definitivamente después de promedias las calificaciones dadas por los usuarios que asistieron al evento. El evento solo puede tener un estado y pertenecer a una categoría, además el organizador puede asociar una o más imágenes a dicho evento.

4.2.10 Diagrama de Estados del Evento. El siguiente diagrama muestra los posibles estados que puede tener un evento con el paso del tiempo.

Figura 21. Diagrama de estados



El evento tiene un estado inicial, **creado**, de allí pasa a ser **publicado** por el usuario. Desde cualquiera de los dos estados anteriores el usuario puede **cancelarlo**, así llega a un estado final.

Si el evento sigue su curso, automáticamente pasa a estado **ejecutado** de acuerdo a la fecha que se especificó cuándo fue creado.

Al ejecutarse o llevarse a cabo el evento, pasarán 4 días para que el evento cambie automáticamente a estado **cerrado**, en este lapso de tiempo los usuarios asistentes al evento podrán calificarlo, una vez cerrado el evento ya no se podrá modificar por ningún tipo de usuario.

4.2.11 Maquetas Iniciales. A continuación, se muestran las maquetas que se hicieron antes del desarrollo.

Figura 22. Maqueta página de portada

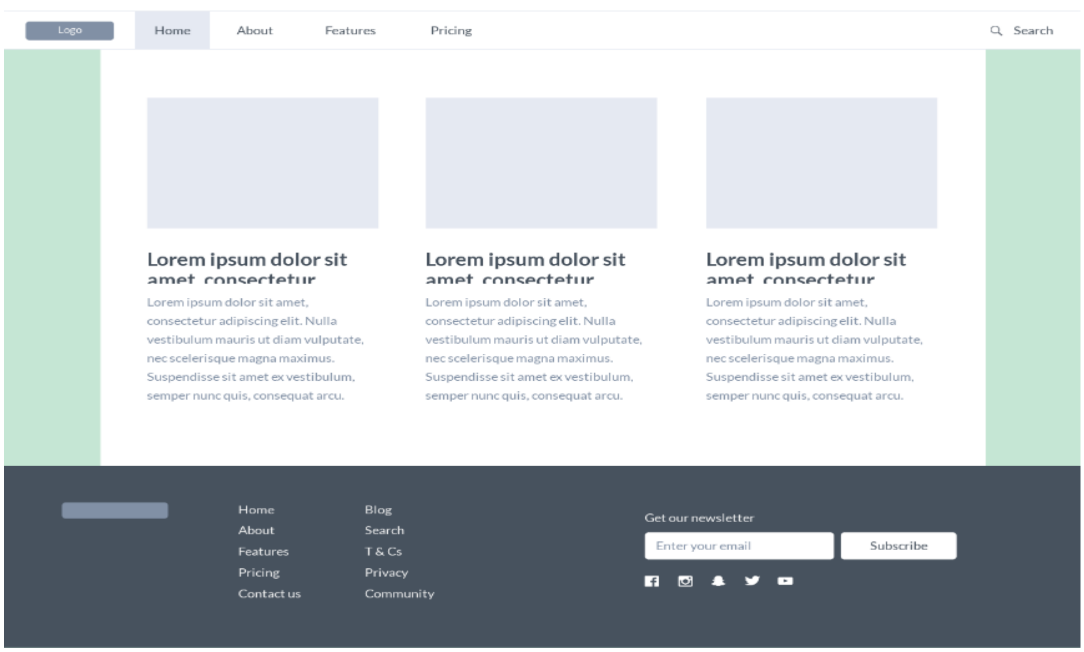


Figura 23. Maqueta página de inicio de la aplicación

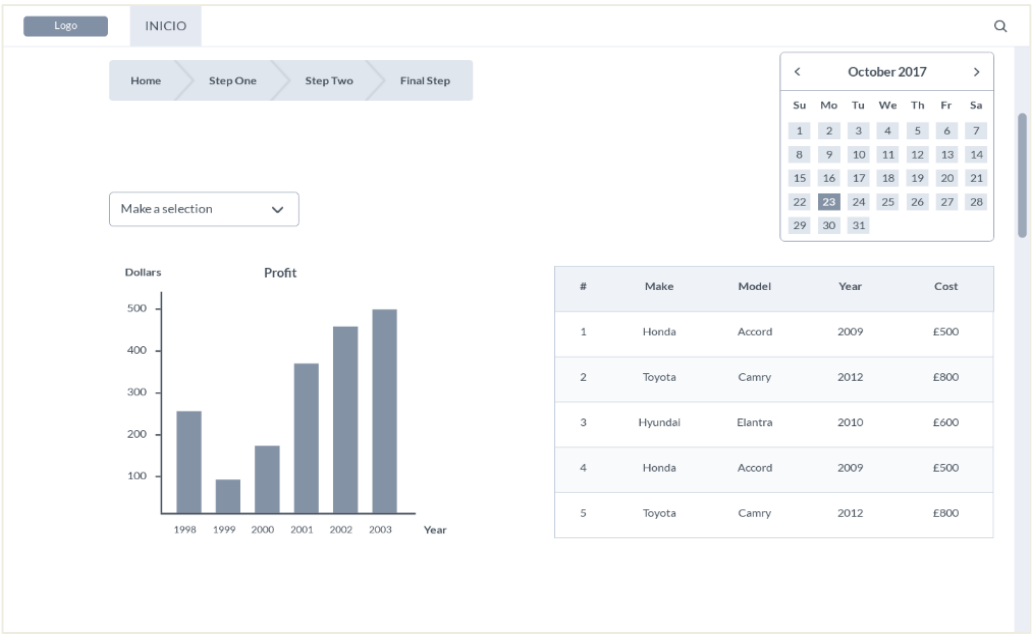
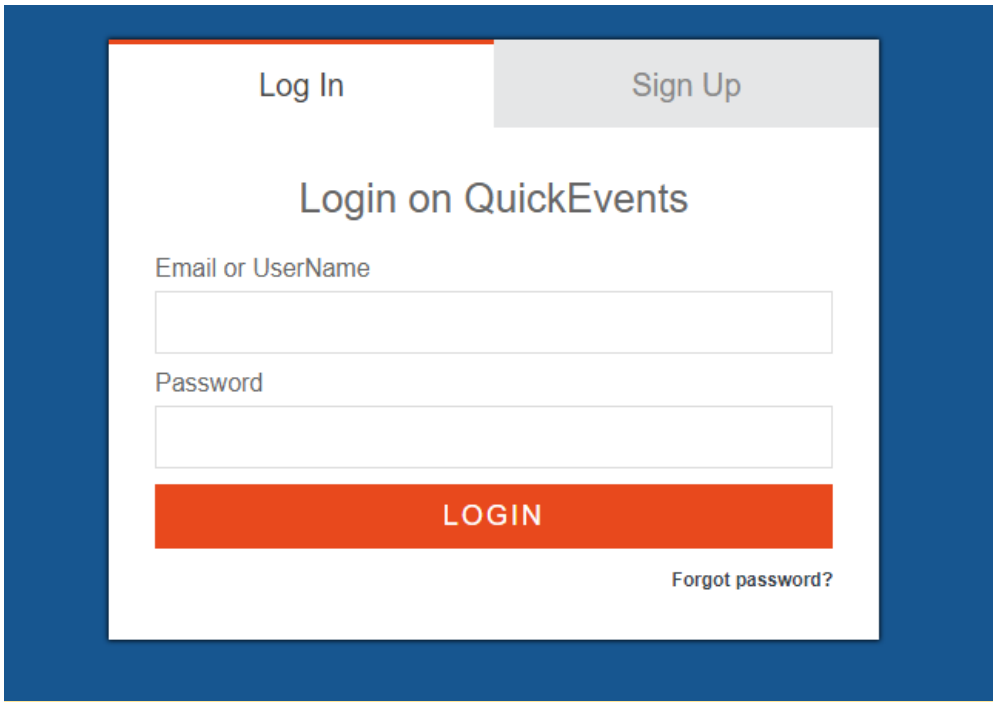


Figura 24. Maqueta con plantilla para dashboard

4.2.12 Prototipos. El desarrollo de la solución tuvo 2 versiones de prototipos los cuales fueron mejorando en la marcha de acuerdo a la visión sobre las funcionalidades finales que se querían ofrecer.

4.2.12.1 Prototipo 1. El primer prototipo se basó en jQuery como biblioteca JavaScript para la manipulación del DOM HTML y solicitudes AJAX a un servidor de pruebas proporcionado por la plataforma Google Apps Script[15] el cual respondía según la petición, y la información provenía de una hoja de cálculo de Google, ésta simulaba la base de datos. Este prototipo permite crear un evento, visualizar lista de eventos y ver detalle de un evento.

Figura 25. Formulario de inicio de sesión, prototipo 1



[15] Overview of Google Apps Script. [En línea]. (Recuperado en 30 de Julio de 2019). Disponible en https://developers.google.com/apps-script/overview

Figura 26. Formulario para crear evento, prototipo 1

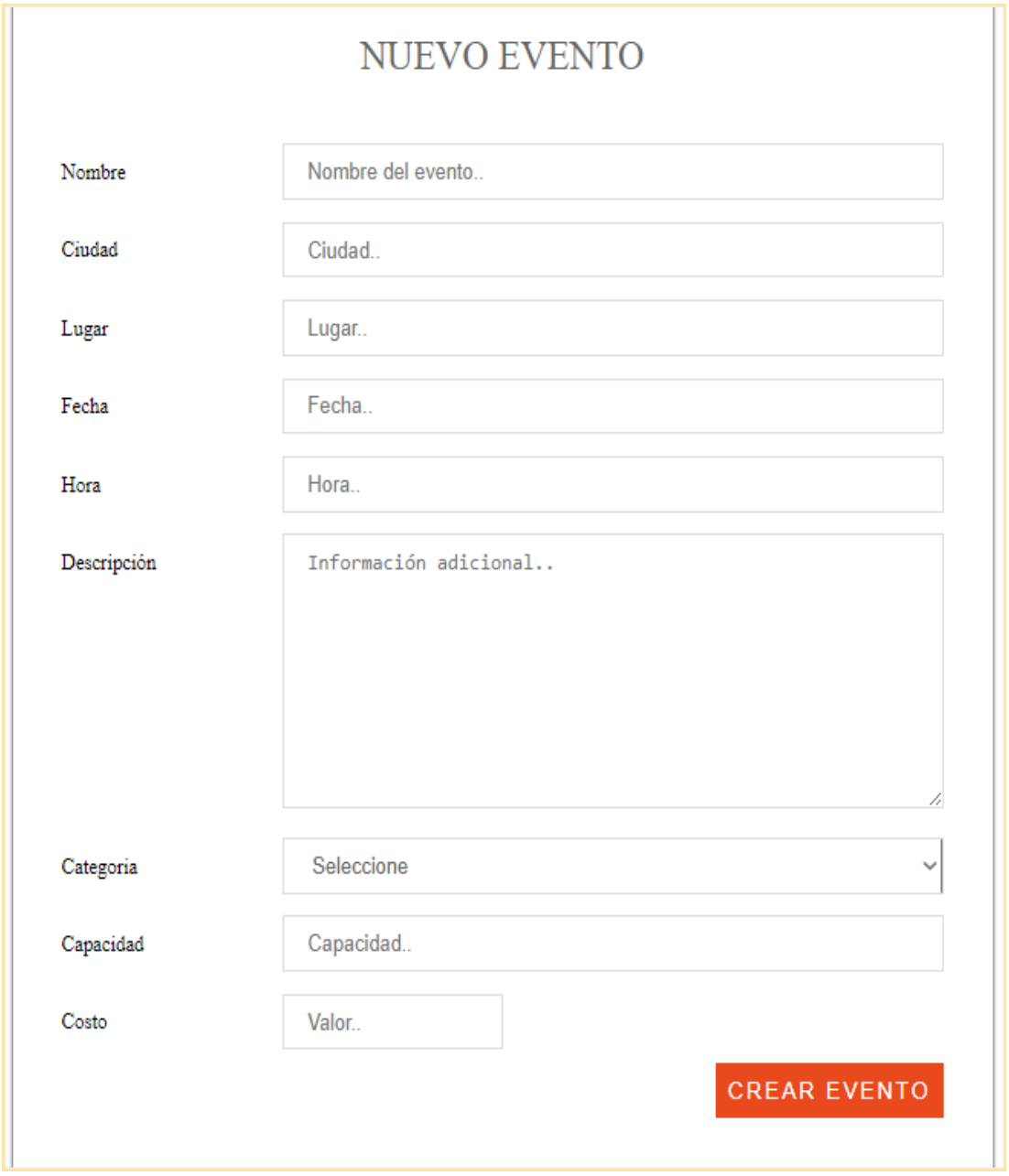


Figura 27. Lista de eventos, prototipo 1

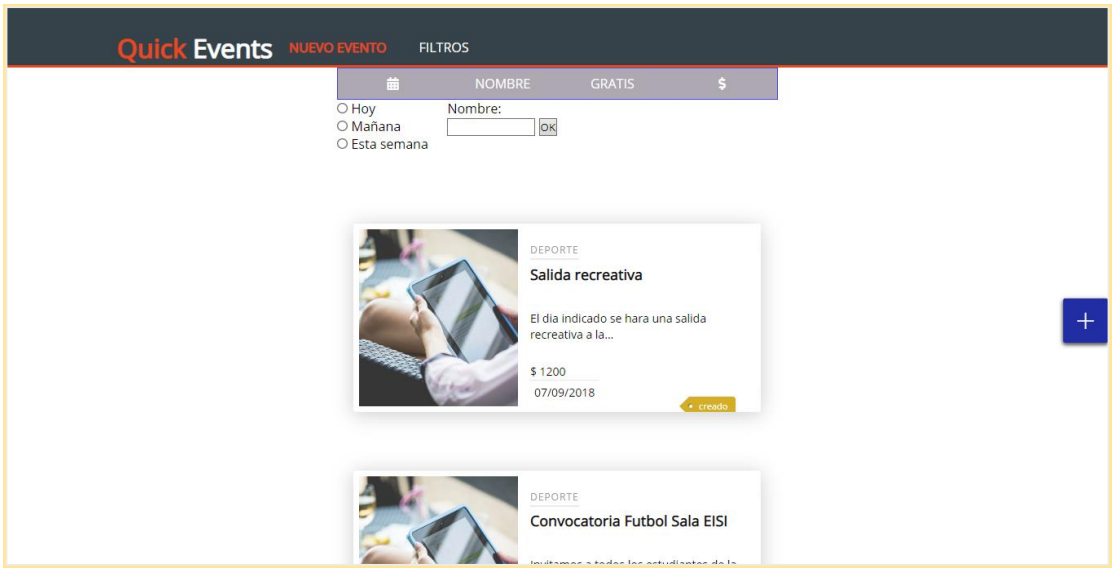
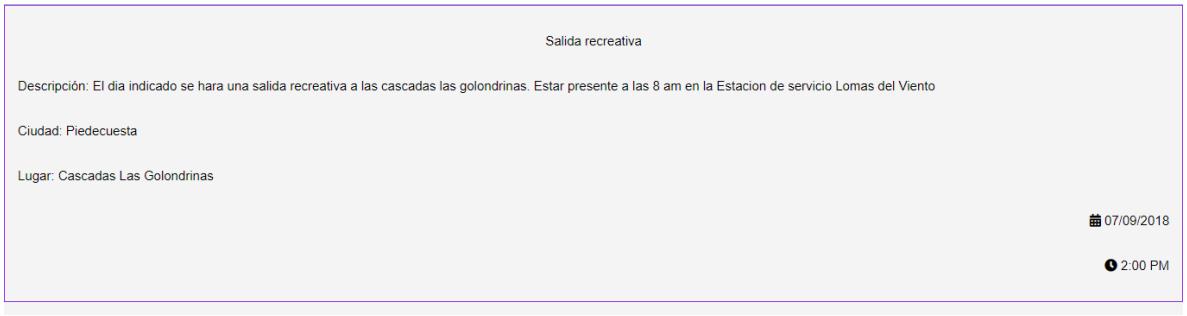


Figura 28. Información de un evento, prototipo 1



4.2.12.2 Prototipo 2. El segundo prototipo fue introducido debido a la necesidad de organizar más adecuadamente el código de la aplicación para, sobre todo, hacer las solicitudes AJAX de manera que en lo posible no se repitiera constantemente el mismo código, y para facilitar la edición y visualización los de eventos.

Como se ha mencionado, el servidor de prueba se hizo en Nodejs. Este backend logró simular todas las operaciones necesarias de la aplicación, haciendo uso también del formato JSON para la comunicación entre cliente y servidor.

Figura 29. Formulario de inicio de sesión y registro, prototipo 2

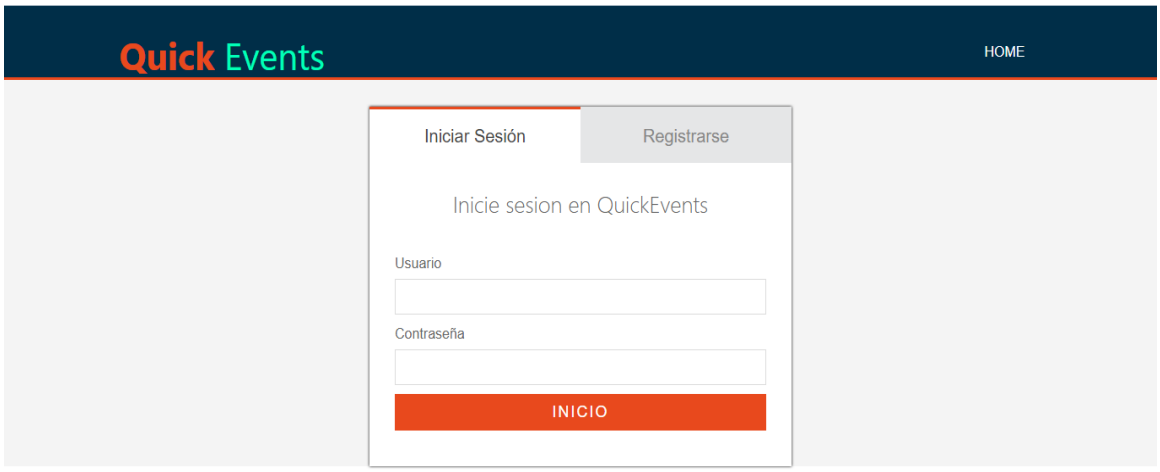


Figura 30. Página de inicio, prototipo 2

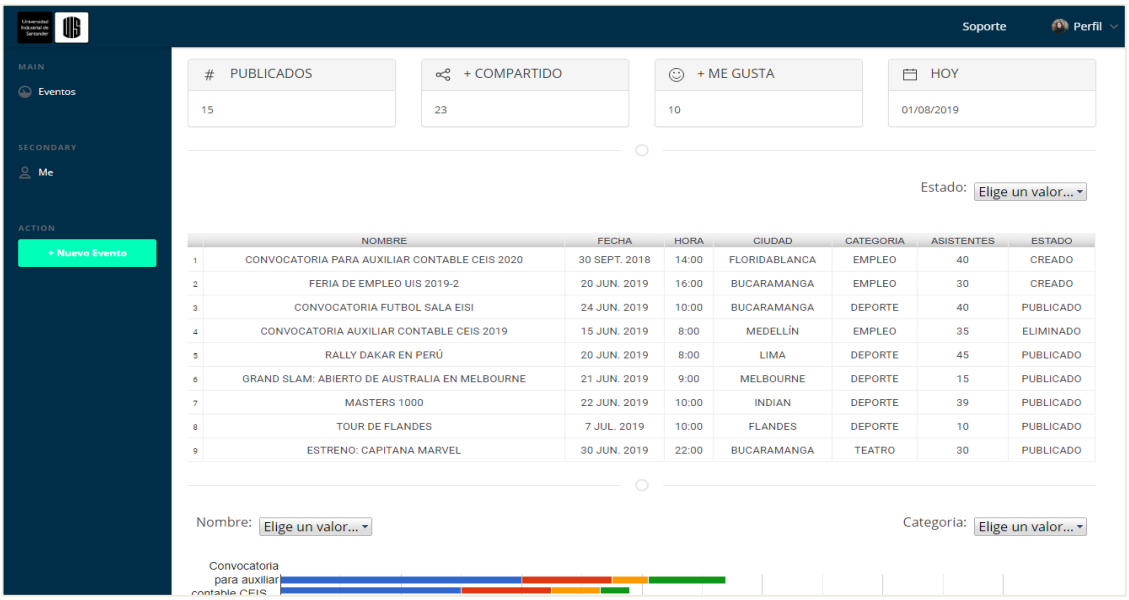


Figura 31. Lista de eventos por estado, prototipo 2

Figura 32. Detalles de un evento, prototipo 2

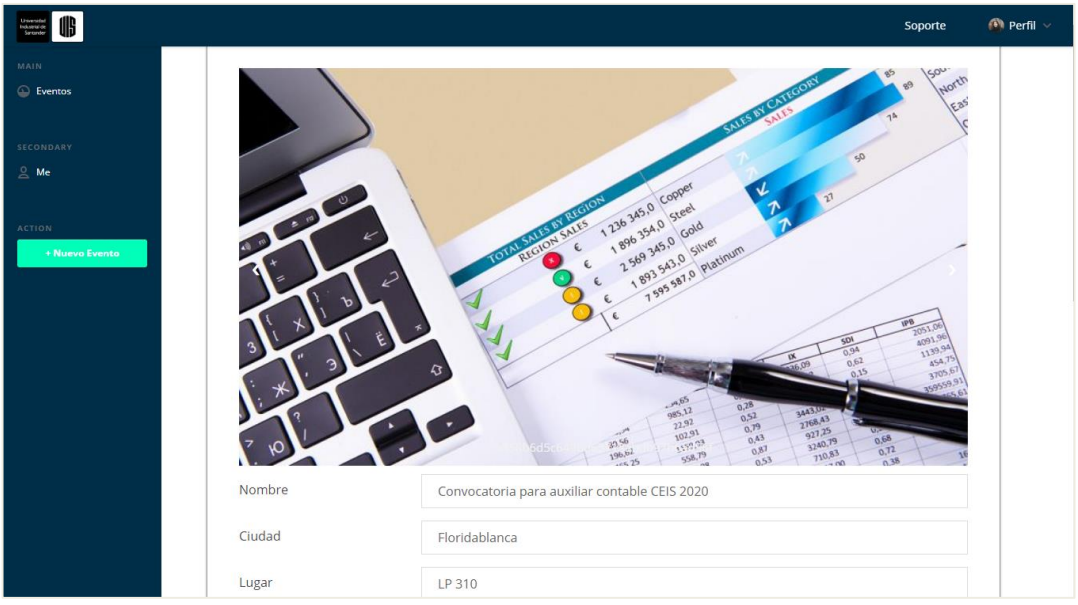


Figura 33. Formulario para crear un evento, prototipo 2

Figura 34. Página de inicio desde dispositivo móvil

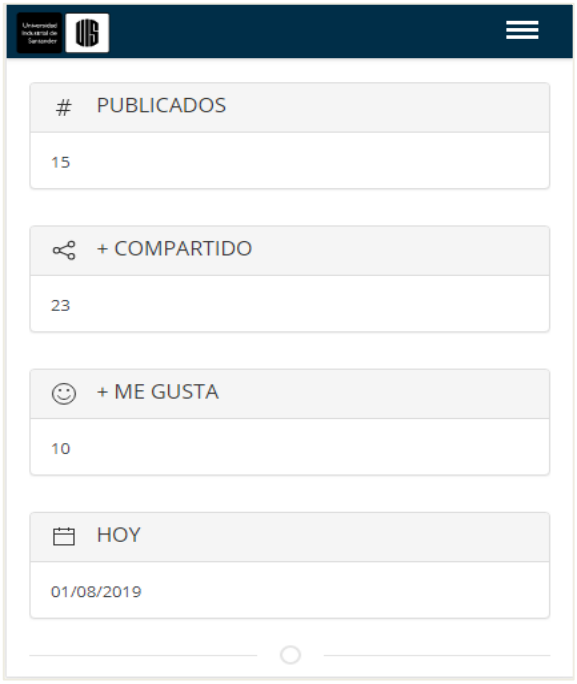


Figura 35. Lista de eventos desde dispositivo móvil

Figura 36. Menú de la aplicación desde dispositivo móvil

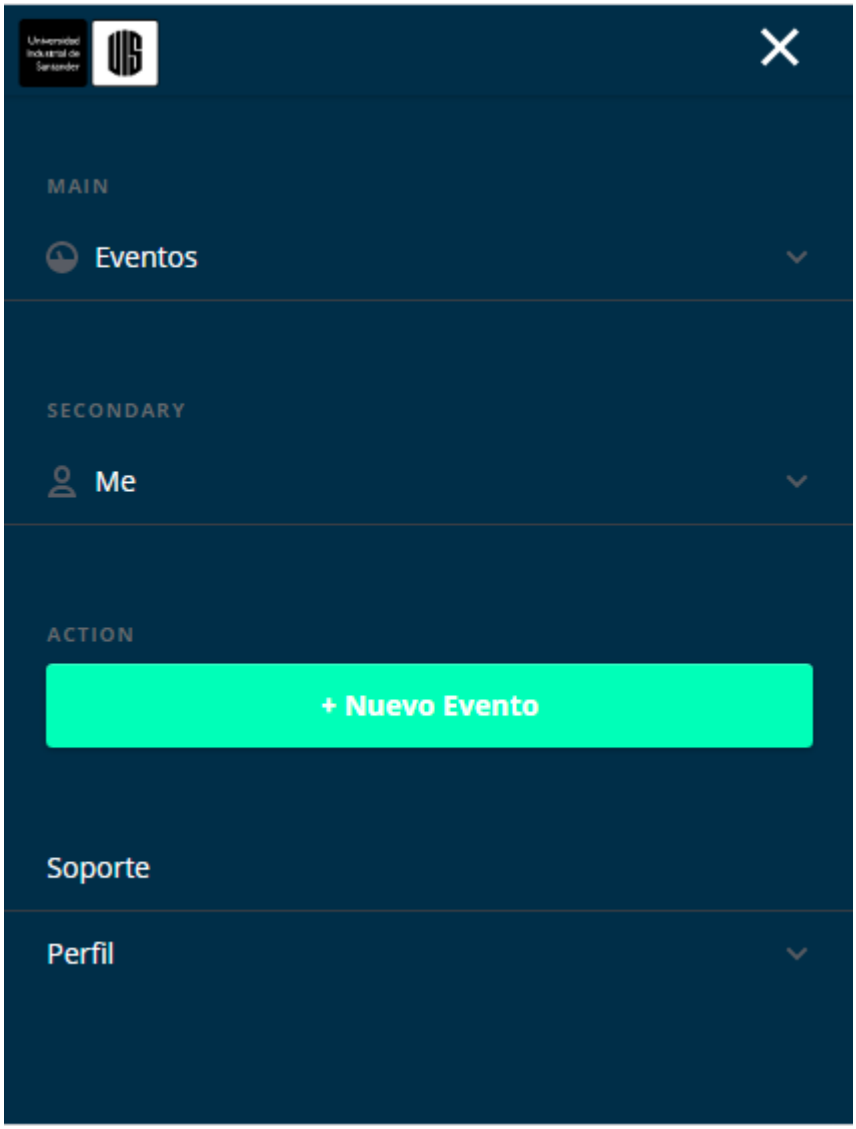


Figura 37. Detalle de un evento desde dispositivo móvil

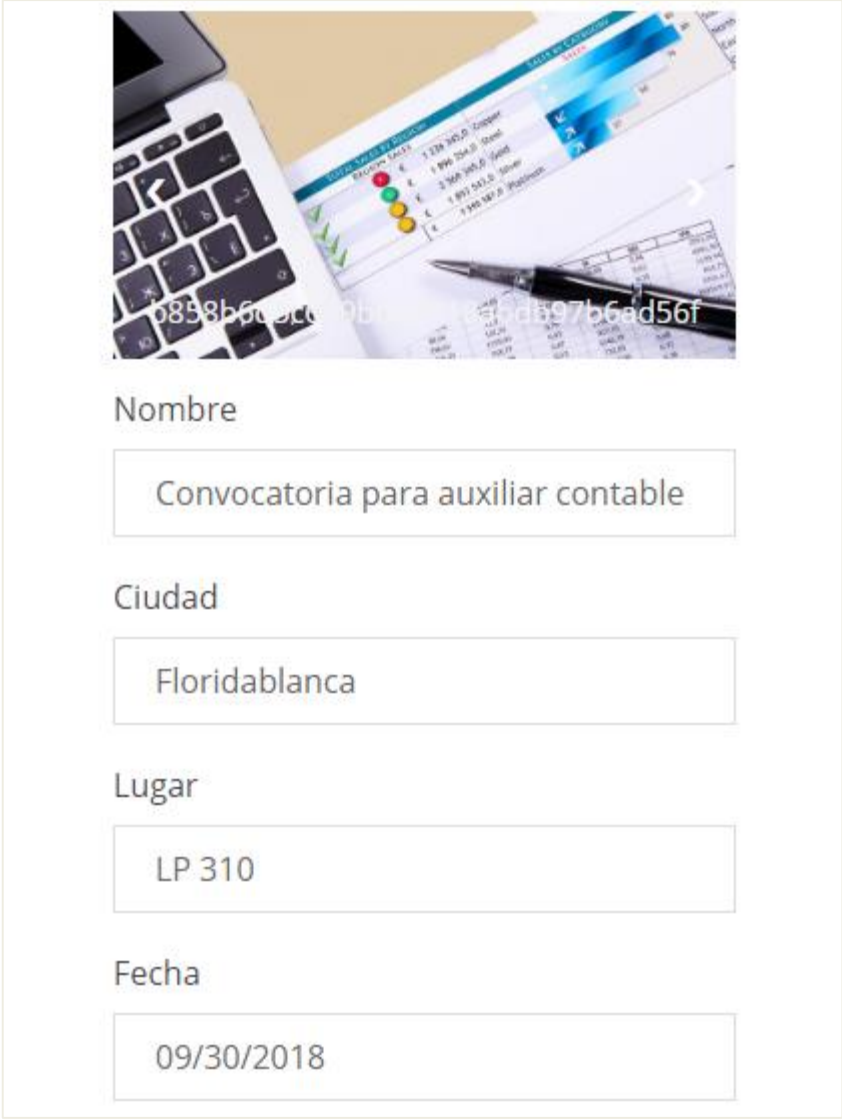
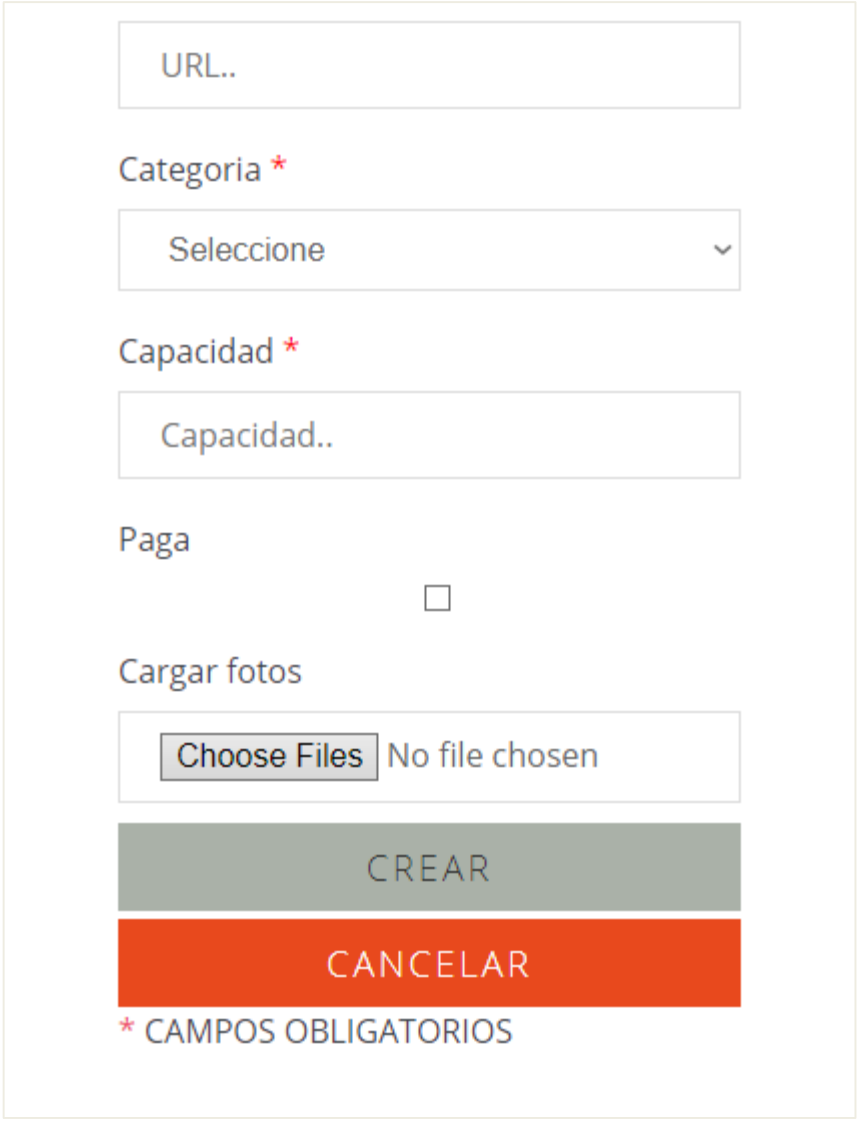


Figura 38. Formulario de registro de evento desde dispositivo móvil -1

Figura 369. Formulario de registro de evento desde dispositivo móvil -2



Las capturas inmediatamente anteriores muestran como la aplicación es adaptable a dispositivos móviles y sus diferentes elementos HTML se ordenan debidamente para responder a cada tamaño desde donde se visualiza. La aplicación web puede encontrarse en su repositorio en GitHub: https://github.com/BrayanG26/my-events-project-backbone. El servidor para probar el funcionamiento de la aplicación puede encontrarse en su repositorio en GitHub: https://github.com/BrayanG26/quick-events-API.

5. VALIDACIÓN

Para la prueba de validación de la usabilidad de la aplicación se realizó una encuesta de satisfacción y comprensión de los procedimientos que ofrece la aplicación. Las preguntas fueron diseñadas para obtener resultados cuantitativos y cualitativos para un mejor análisis y comprender qué posibles mejoras son necesarias a futuro.

Tabla 16. Encuesta

PREGUNTAS	OPCIONES
1. ¿Cuán satisfecho se encuentra con la página de inicio de la aplicación?	(1) Muy insatisfecho (2) Insatisfecho (3) Neutral (4) Satisfecho (5) Muy satisfecho
2. ¿Cada elemento de la tarjeta de eventos le dio los detalles que necesitaba conocer sobre dicho evento?	(a) SI (b) NO
3. ¿Es útil la información que proporcionan los gráficos sobre los eventos en la pantalla de inicio?	(1) Nada útil (2) Poco útil (3) Útil (4) Muy útil
4. ¿Ha logrado filtrar los eventos por estado?	(a) SI (b) NO
5. ¿Ha logrado filtrar los eventos por categoría?	(a) SI (b) NO
6. ¿Qué tan fácil fue modificar un evento?	(1) Muy difícil (2) Difícil (3) Poco fácil (4) Fácil (5) Muy fácil
7. ¿Cuán satisfecho se encuentra con la visualización de las imágenes en los detalles de un evento?	(1) Muy insatisfecho (2) Insatisfecho (3) Neutral (4) Satisfecho (5) Muy satisfecho
8. ¿Cuán fácil fue cargar nuevas imágenes a un evento?	(1) Muy difícil (2) Difícil (3) Poco fácil (4) Fácil (5) Muy fácil
9. ¿Cuán fácil fue crear un evento?	(1) Muy difícil (2) Difícil (3) Poco fácil (4) Fácil (5) Muy fácil
10. ¿Usaría esta aplicación web en el futuro?	(a) SI (b) NO
11. ¿Tiene alguna sugerencia? ¿Qué funciones adicionales le gustaría que tenga la aplicación?	Respuesta:

5.1 RESULTADOS DE LAS PRUEBAS DE VALIDACIÓN

Los resultados de las pruebas de validación se pueden encontrar en el anexo A. A continuación, se muestran en forma de gráficos los resultados de cada pregunta a un grupo de seis personas.

Figura 40. Pregunta 1 - Encuesta

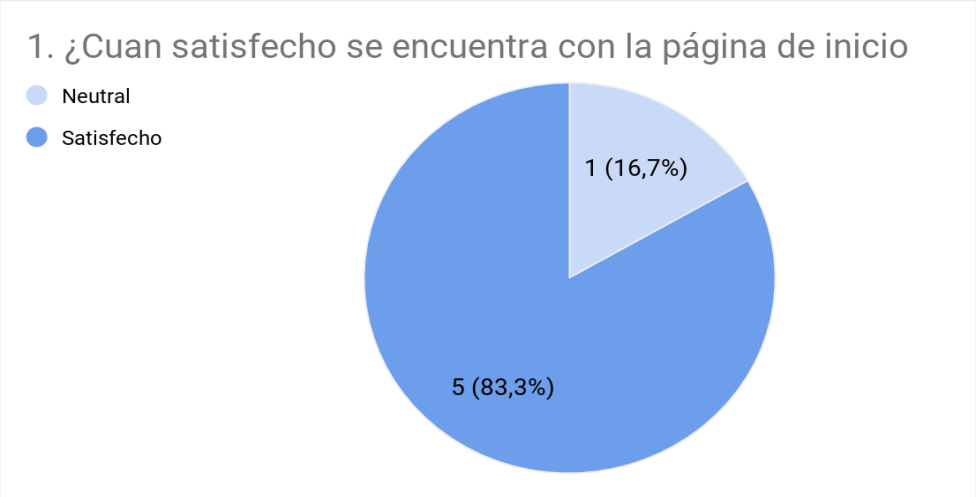


Figura 41. Pregunta 2 - Encuesta

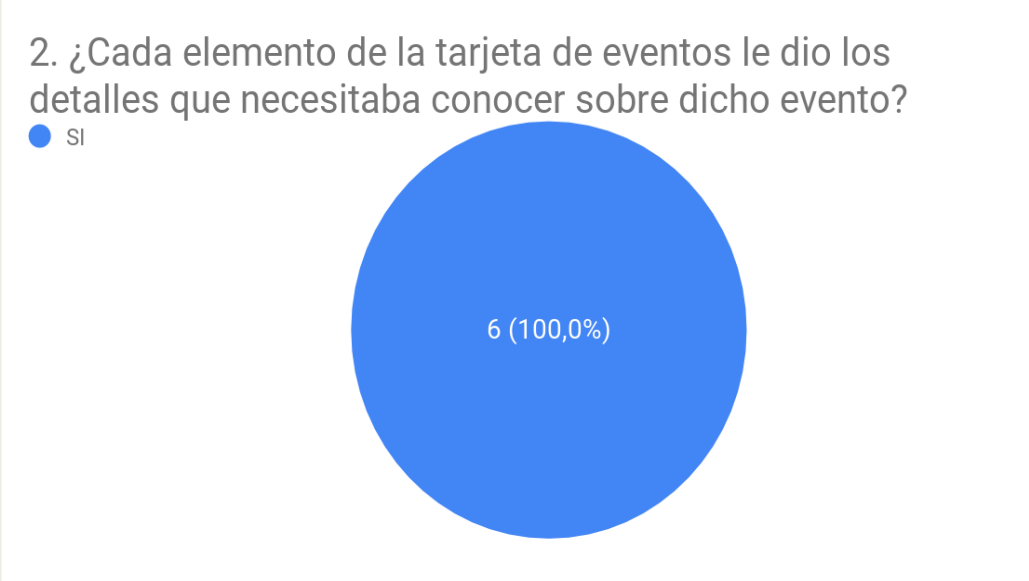


Figura 42. Pregunta 3 - Encuesta

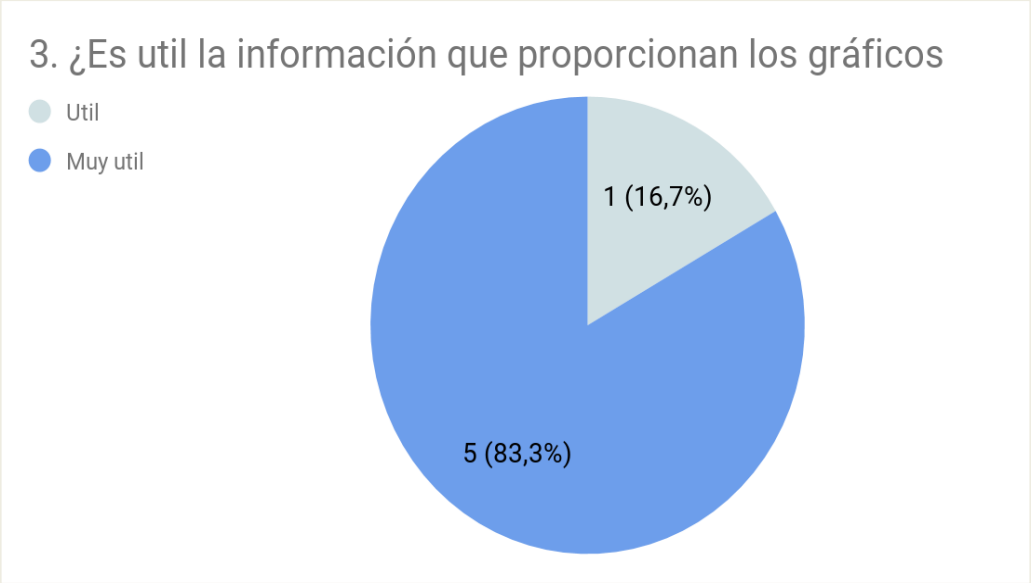


Figura 43. Pregunta 4 - Encuesta

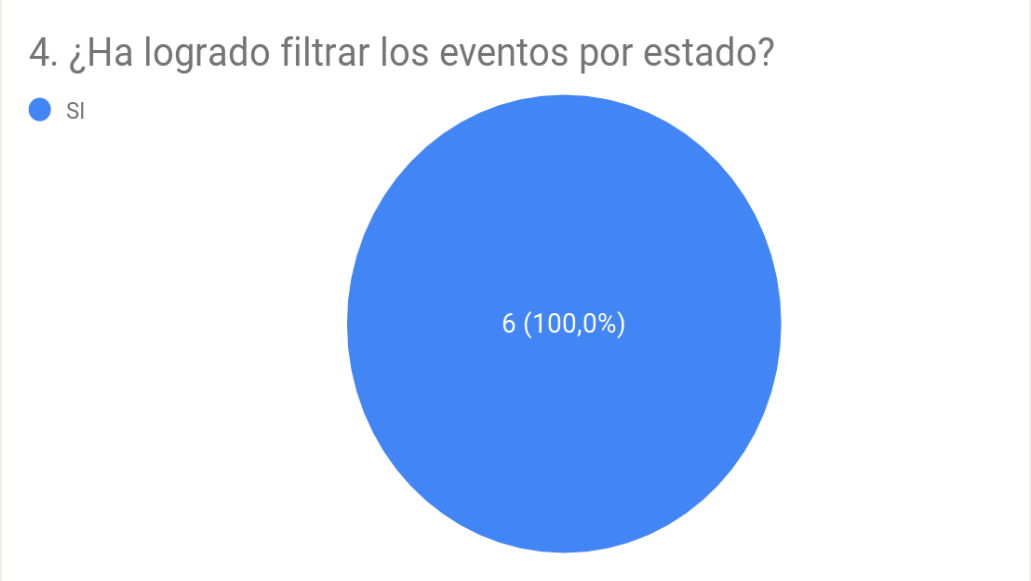


Figura 44. Pregunta 5 - Encuesta

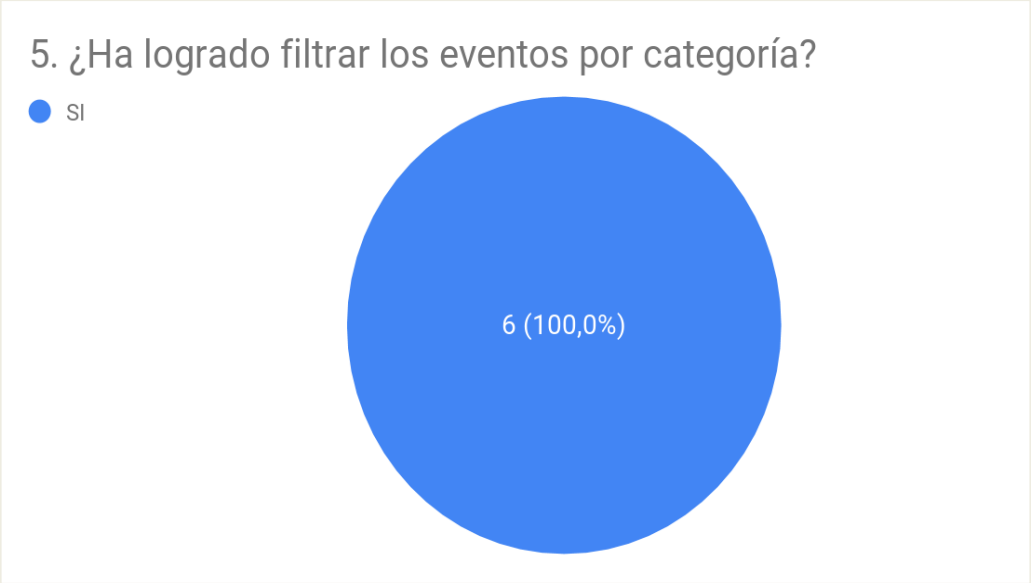


Figura 45. Pregunta 6 - Encuesta

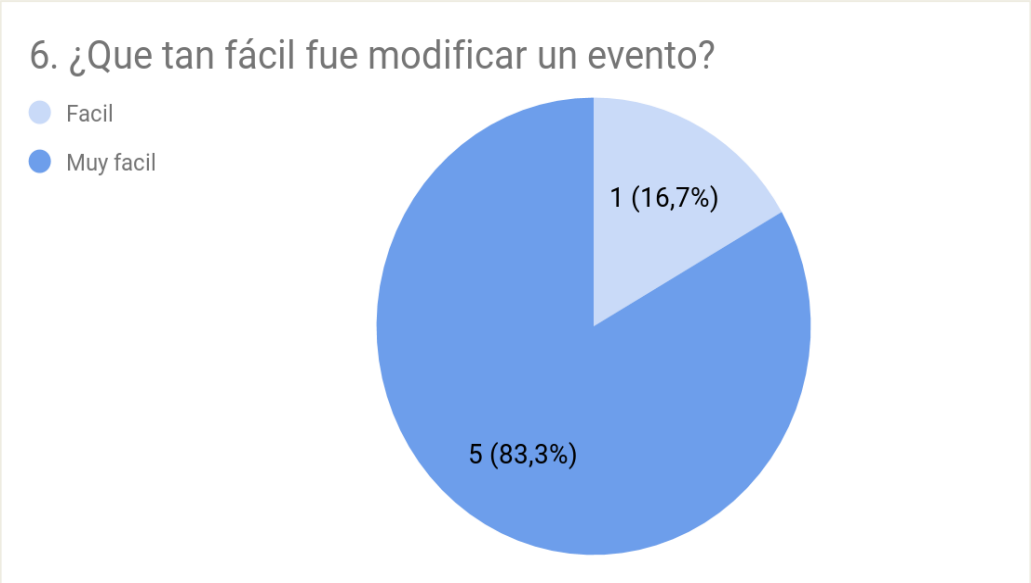


Figura 46. Pregunta 7 - Encuesta

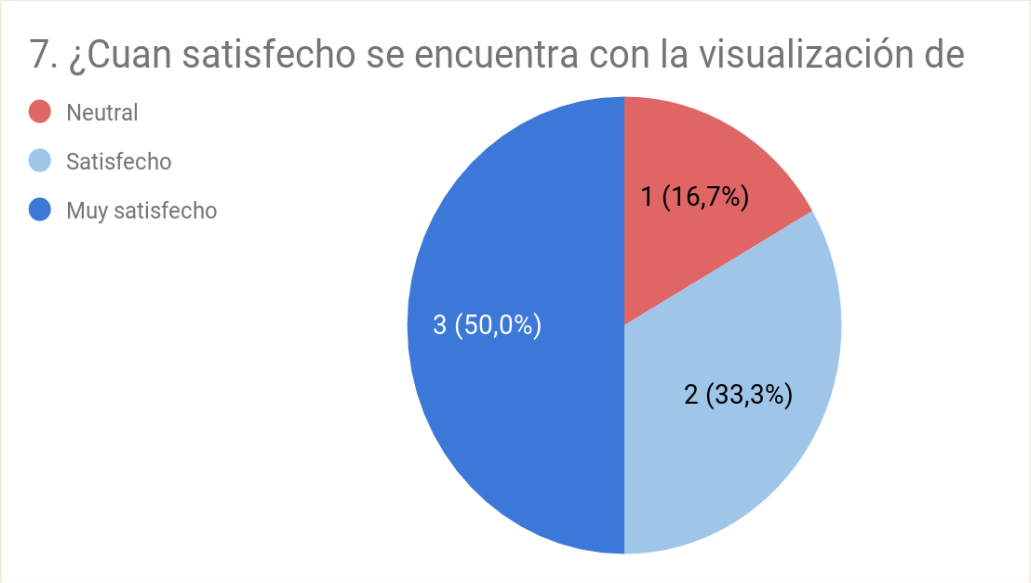


Figura 47. Pregunta 8 - Encuesta

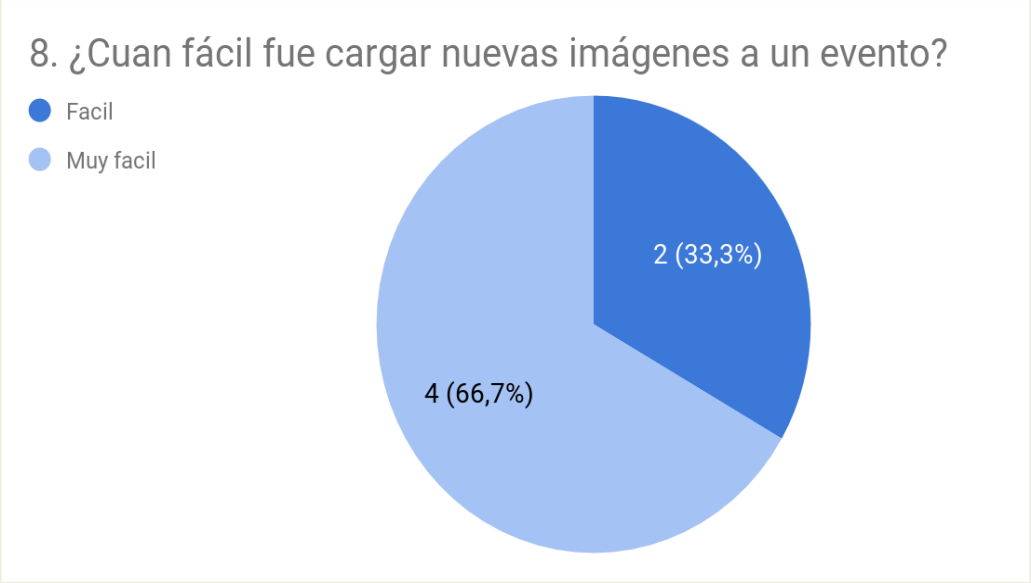


Figura 48. Pregunta 9 - Encuesta

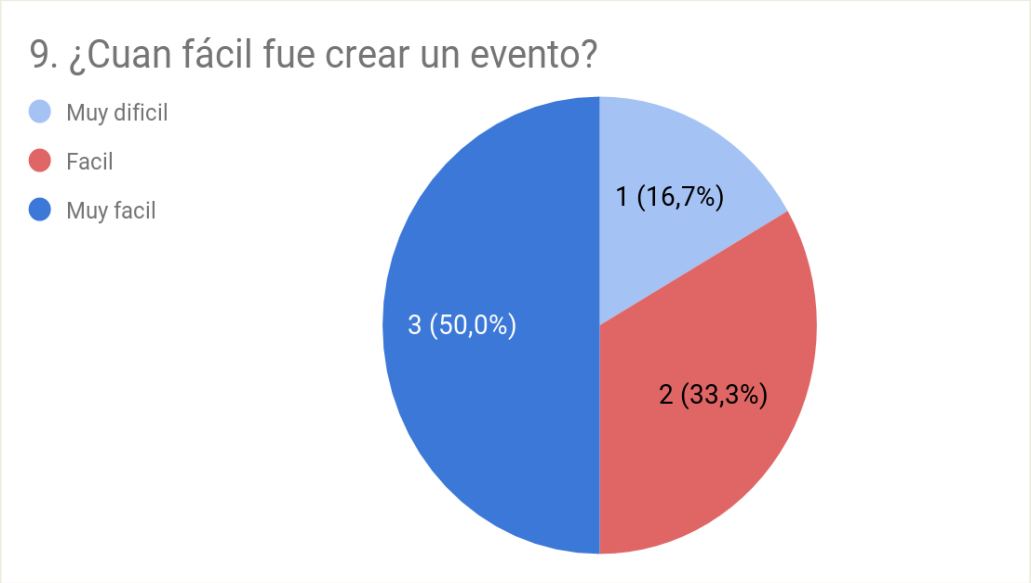
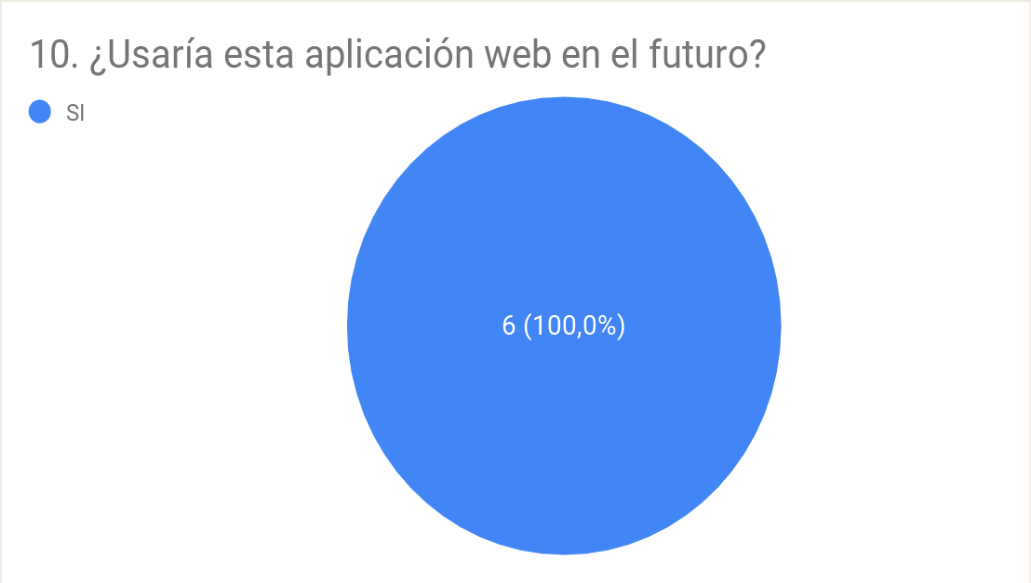


Figura 49. Pregunta 10 - Encuesta



Los resultados que arrojó la encuesta permiten inferir que la aplicación en general fue aceptada por los encuestados dado que según manifestó la mayoría, es un producto funcional de lo que se espera de una aplicación de este tipo. Sin embargo, la mayoría también mostró una disposición similar en que aún hacían falta ciertas

características que permitan al usuario un poco más de intuitividad al momento de usar la aplicación cuando se encaran por primera vez a manejarla y no saben nada de ella.

6. CONCLUSIONES

Mediante el uso de esta plataforma conjunta, aplicación móvil, y aplicación web es factible complementar los canales de difusión de información de eventos.

Se ha implementado un mecanismo para visualizar los atributos que los usuarios (desde aplicación móvil) modifican, y que permite calificar un evento, estos son: logística, interesante, entretenido, comodidad.

El prototipo fue desarrollado enfocado también para acceder desde dispositivos móviles, esto aumenta la facilidad de uso, al permitir que el organizador o administrador cree rápidamente un evento nuevo desde un smartphone o tablet.

Se ha logrado desarrollar un prototipo funcional que cumple con los requisitos funcionales planteados.

Se ha logrado desarrollar una aplicación web usando un conjunto de tecnologías front-end que permite al organizador la creación y publicación de eventos en cinco categorías básicas: *empleo, académico, música, teatro, deporte*.

Teniendo en cuenta la razón de ser de este proyecto - unificar canales de difusión de eventos y agilizar el proceso de difusión -, esto es una tarea que requiere la infraestructura física para tener la disponibilidad de información precisa en el momento oportuno para ser fiable al momento de mostrar al usuario.

Mediante este proyecto hemos podido apreciar que aprovechar la información que se genera en la universidad mediante herramientas como ésta, es algo valioso para el progreso de una comunidad como la nuestra. Esto ciertamente es de gran ayuda con miras al futuro, teniendo en cuenta que como profesionales podemos generar

cierto impacto en el progreso del país, y en este campo de la Ingeniería de Sistemas, las *Smart Cities*, hacia allí nos vamos encaminando.

7. TRABAJO FUTURO

El proyecto desarrollado ha dejado como enseñanza la viabilidad en la implementación de este tipo de recursos. Permite agilizar y mejorar el flujo de la información. No obstante, también hay margen para el mejoramiento, por ejemplo:

- Es posible implementar el uso de etiquetas para los eventos, esto permitirá filtrar mejor la información y buscar un evento.
- Hacer uso de un framework CSS permitirá unificar los estilos de toda la plataforma y mejorar la calidad de la visualización no solo desde computadores de escritorio, sino también desde dispositivos móviles.
- En caso de ser pertinente, es posible hacer una mezcla para la visualización de gráficos, usando un plugin JavaScript pertinente junto con CSS, esto con el objetivo de alivianar la carga de procesamiento al momento de que el navegador renderiza la página.
- La carga de imágenes puede ser mejorada implementando una carga retardada de las mismas en lugar de hacer peticiones al mismo tiempo de todas las imágenes visibles en la página, esto dará una impresión de rapidez al usuario.
- En caso de ser necesario hacer una reingeniería de la aplicación web, se recomienda usar Angular o Vue.js como frameworks JavaScript debido a que estos cuentan con una amplia documentación que Backbone.js.
- Se puede mejorar la autenticación durante el proceso de inicio de sesión de un usuario, y a lo largo de todas las peticiones que hace la aplicación, usando un *token* como método de seguridad para las peticiones HTTP.
- Se podría implementar la funcionalidad *drag and drop* cuando se intenta modificar o agregar las imágenes asociadas a un evento.
- En una siguiente versión es conveniente analizar las mejoras sugeridas por los encuestados en esta etapa para facilitar la navegación a través de la aplicación y hacer el trabajo más fácil para el administrador de eventos.

BIBLIOGRAFÍA

BOUSKELA, Mauricio; CASSEB, Marcia; BASSI, Silvia; DE LUCA, Cristina; FACCHINA Marcelo. La ruta hacia las Smart Cities: Migrando de una gestión tradicional a la ciudad inteligente. (Monografía del BID; 454), 2016, 148p.

DISEÑO DE API. [En línea]. (Recuperado en 30 de Julio 2019). Disponible en https://usedsources.page.link/fFWD

DOGLIO Fernando. REST API Development with Node.js. Apress, 2018. ISBN-13 (pbk): 978-1-4842-3714-4 ISBN-13 (electronic): 978-1-4842-3715-1

ECHAMEA Abiee, Mastering Backbone.js. Packt Publishing, 2016. p.1-16. ISBN 978-1-78328-849-6

GOURLEY David, TOTTY Brian. HTTP: The Definitive Guide. O'Reilly Media, 2002 p.43. ISBN-10: 1-56592-509-2 ISBN-13: 978-1-56592-509-0

NIELSEN Jakob, Usability Engineering. Academic Press, 1993. ISBN-13: 978-0-12-518406-9 ISBN-10: 0-12-518406-9

OSMANI Addy, Developing Backbone.js Applications. O'Reilly Media, 2013. ISBN 978-1-449-32825-2

MIRGOROD Vadim, Backbone.js Cookbok. Packt Publishing, 2013. p.54-57. ISBN 978-1-78216-272-8

WALKER Jeremy, Backbone.js Essentials. Packt Publishing, 2015. p.36-64. ISBN 978-1-78439-479-0

ANEXOS

Anexo A. Resultados de los usuarios evaluados

Respuestas a preguntas de opción múltiple

PREGUNTA	1	2	3	4	5	6
1. ¿Cuán satisfecho se encuentra con la página de inicio de la aplicación?	(3) Neutral	(4)Satisfecho	(4)Satisfecho	(4)Satisfecho	(4)Satisfecho	(4)Satisfecho
2. ¿Cada elemento de la tarjeta de eventos le dio los detalles que necesitaba conocer sobre dicho evento?	(a)Si	(a)Si	(a)Si	(a)Si	(a)Si	(a)Si
3. ¿Es útil la información que proporcionan los gráficos sobre los eventos en la pantalla de inicio?	(4)Muy útil	(4)Muy útil	(3)Útil	(4)Muy útil	(4)Muy útil	(4)Muy útil
4. ¿Ha logrado filtrar los eventos por estado?	(a)Si	(a)Si	(a)Si	(a)Si	(a)Si	(a)Si
5. ¿Ha logrado filtrar los eventos por categoría?	(a)Si	(a)Si	(a)Si	(a)Si	(a)Si	(a)Si
6. ¿Que tan facil fue modificar un evento?	(5)Muy fácil	(5)Muy fácil	(4)Fácil	(5)Muy fácil	(5)Muy fácil	(5)Muy fácil

PREGUNTA	1	2	3	4	5	6
7. ¿Cuán satisfecho se encuentra con la visualización de las imágenes en los detalles de un evento?	(5)Muy satisfecho	(5)Muy satisfecho	(3)Neutral	(4)Satisfecho	(4)Satisfecho	(5)Muy satisfecho
8. ¿Cuán fácil fue cargar nuevas imágenes a un evento?	(5)Muy fácil	(5)Muy fácil	(4)Fácil	(5)Muy fácil	(4)Fácil	(5)Muy fácil
9. ¿Cuán fácil fue crear un evento?	(5)Muy fácil	(4)Fácil	(4)Fácil	(5)Muy fácil	(5)Muy fácil	(1)Muy difícil
10. ¿Usaría esta aplicación web en el futuro?	(a)Si	(a)Si	(a)Si	(a)Si	(a)Si	(a)Si

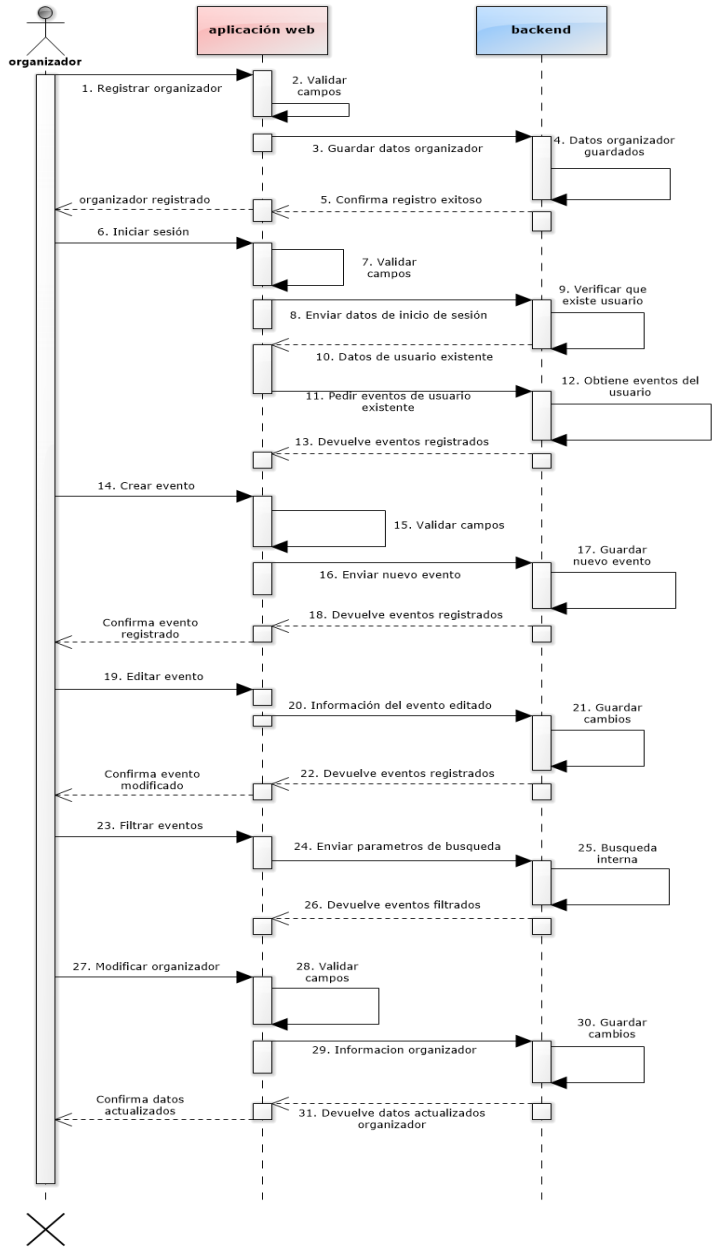
Sugerencias y funciones adicionales

¿Tiene alguna sugerencia?	¿Qué funciones adicionales le gustaría que tuviera la aplicación?
Me gustaría que fuera más intuitiva y amigable con el usuario final. Que los colores sean más vivos y el panel izquierdo se oculte o se minimice. Organizar mejor la pantalla de inicio.	Categorizar mejor la página de inicio
Mejorar diseño y colores	Poner banners a lo largo de la página de inicio con imágenes de eventos recientes y relevantes
Desplegar los gráficos de los eventos con alguna opción desde el panel izquierdo o dando clic en la tabla donde se listan todos los eventos	Ofrecer más funciones desde la tabla de google charts
Organizar las tarjetas para visualizar más en un solo scroll	Indicar el lugar en la tarjeta del evento
Al cargar las imágenes se ofrezca información de carga y éxito al hacerlo. Más interacción con la tabla de google charts	Mantener una conexión en vivo o refrescar cambios cada cierto periodo de tiempo
Ver listado de eventos en una vista independiente. Y las tarjetas de los eventos deberían ser más pequeñas.	

Anexo B. Diagramas del funcionamiento del prototipo de la aplicación

A continuación, se muestran algunos diagramas que complementan el entendimiento del funcionamiento del prototipo desarrollado.

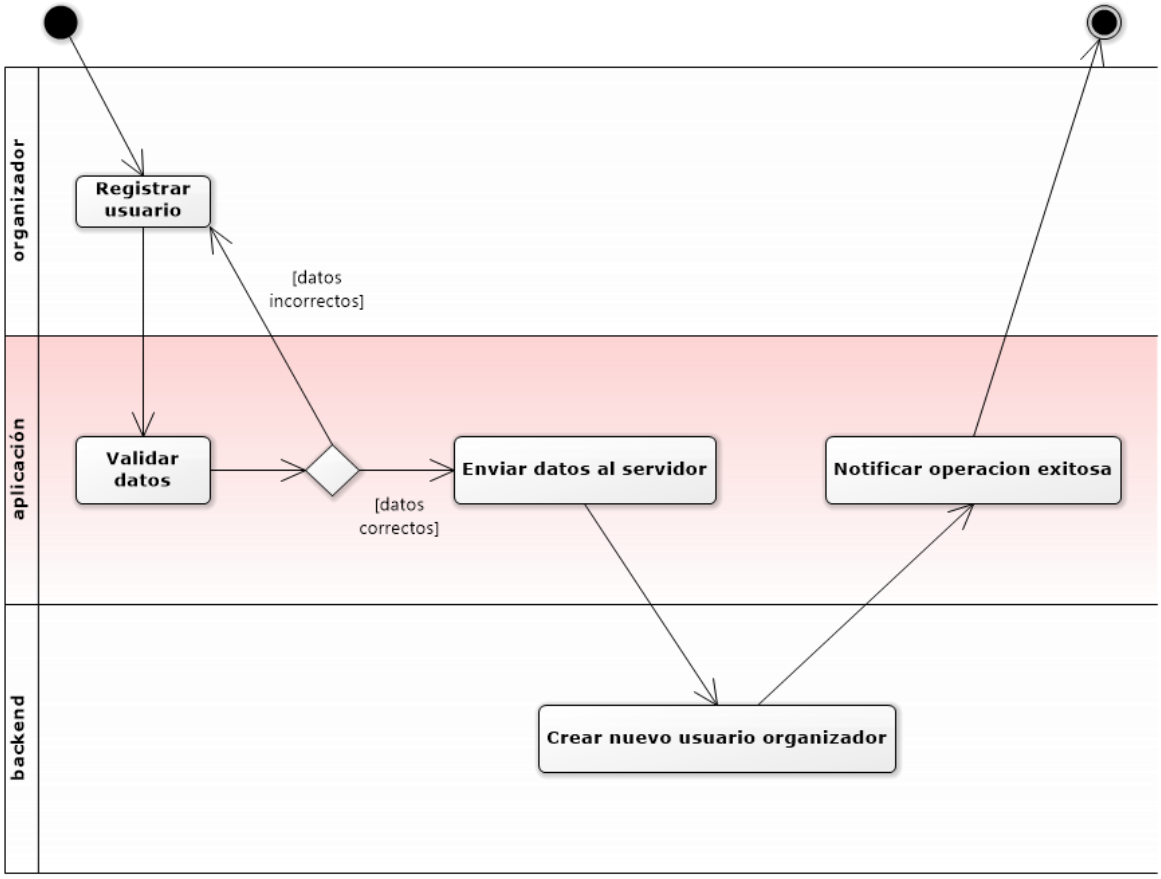
Diagrama de secuencias de la aplicación



El anterior diagrama de secuencias muestra las acciones que realiza cada parte involucrada del sistema: **organizador**, **aplicación** y **backend**. Allí aparecen las funcionalidades que ofrece la aplicación y cómo interactúan las tres partes en cada caso.

A continuación, se muestran los diagramas de actividad que muestra con más detalle las acciones de la figura 48.

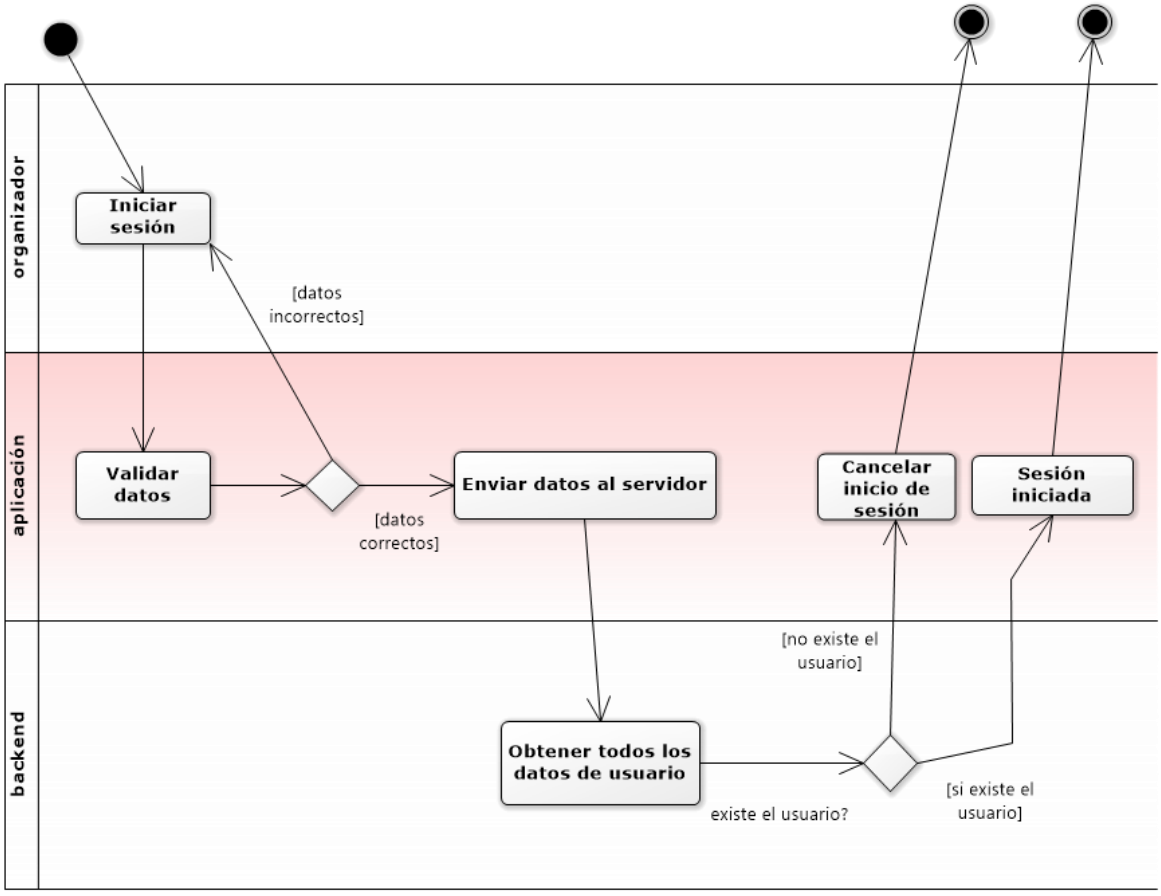
Actividad - Registrar usuario



La figura 49 muestra el procedimiento para registrar un usuario. La aplicación valida los datos ingresados por el usuario, si son válidos los envía al servidor. Este crea

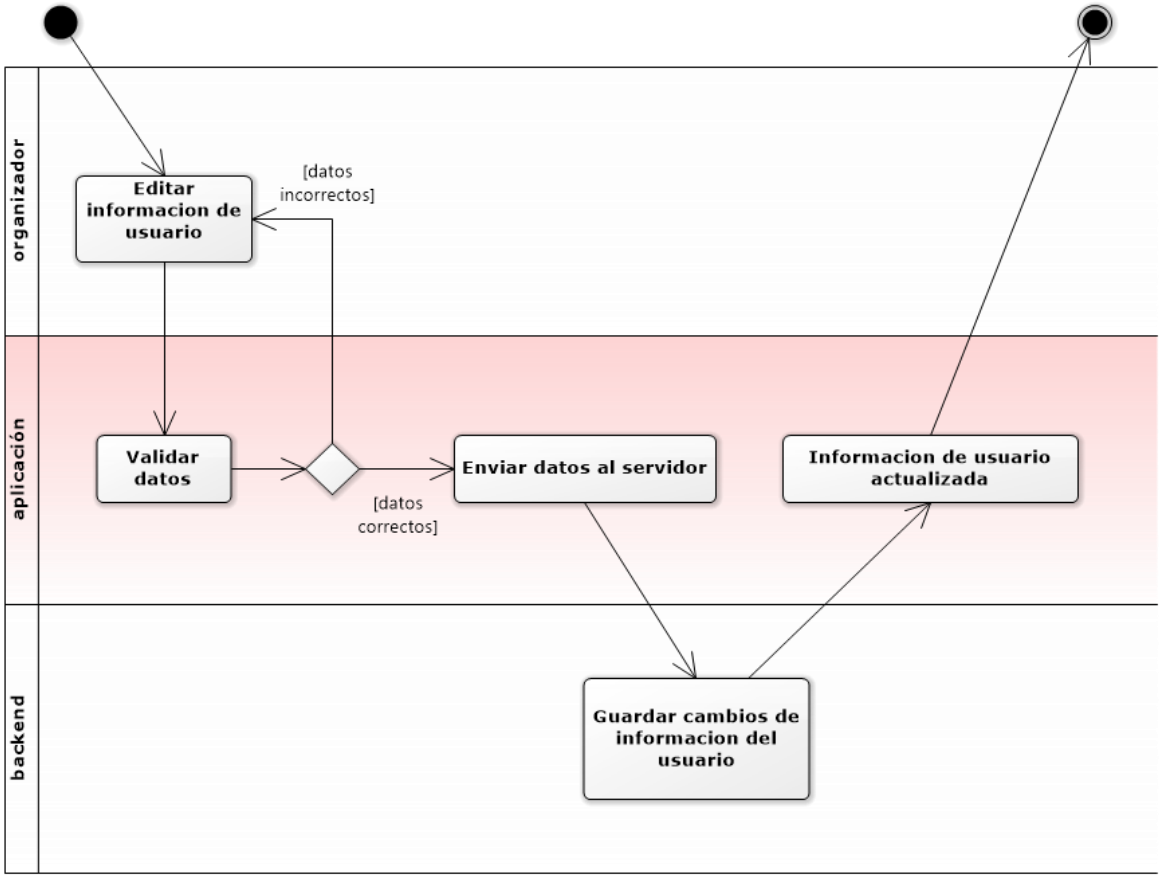
un nuevo usuario que se guarda en base de datos y posteriormente confirma la creación de dicho usuario.

Actividad - Iniciar sesión



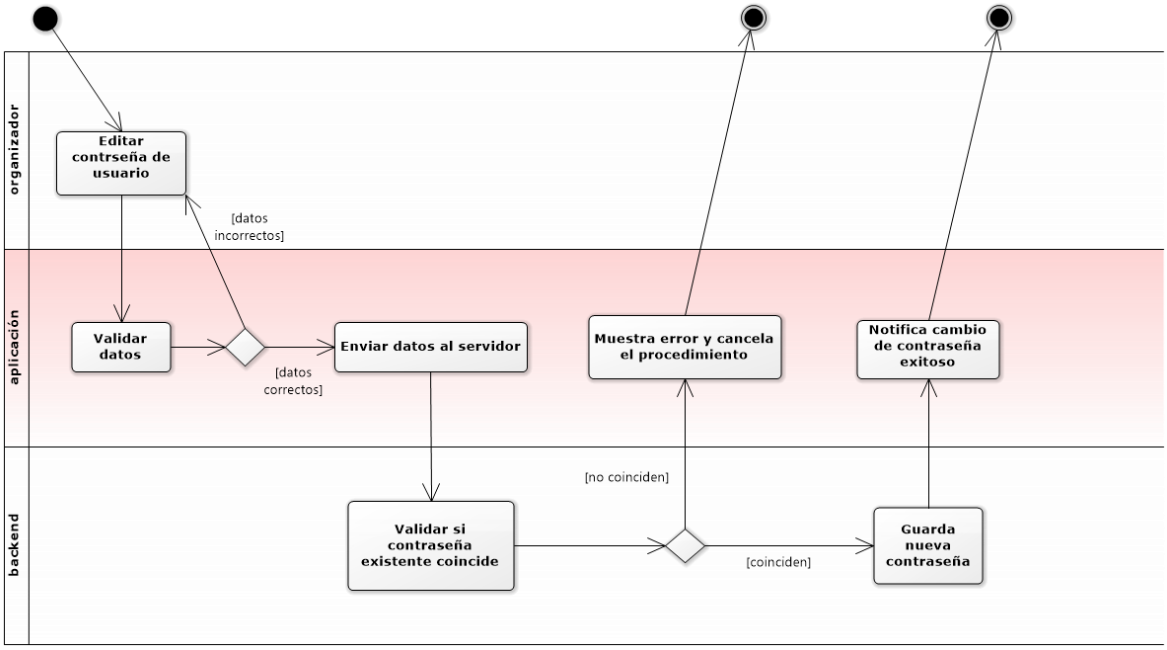
La figura 50 muestra el procedimiento para iniciar sesión después de haberse registrado el usuario. La aplicación valida los datos ingresados para que no haya caracteres indeseados. Si es válida la información se envía al servidor. Allí se busca el usuario ingresado, en caso de ser encontrado, los datos son recibidos por la aplicación. En caso contrario, se le notificara al usuario que hubo un error.

Actividad - Editar información de usuario



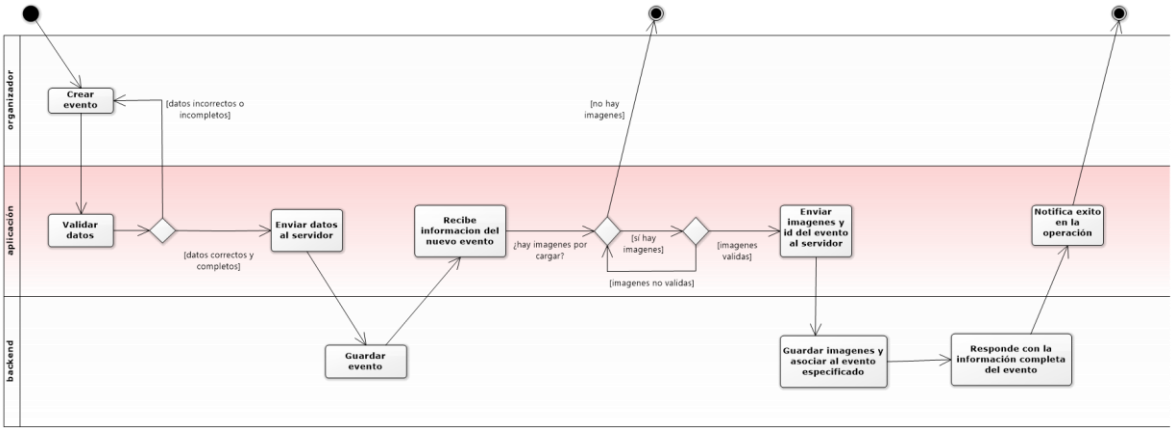
La figura 50 muestra el procedimiento para editar la información del usuario, pero esto no incluye la contraseña.

Actividad - Editar contraseña de usuario



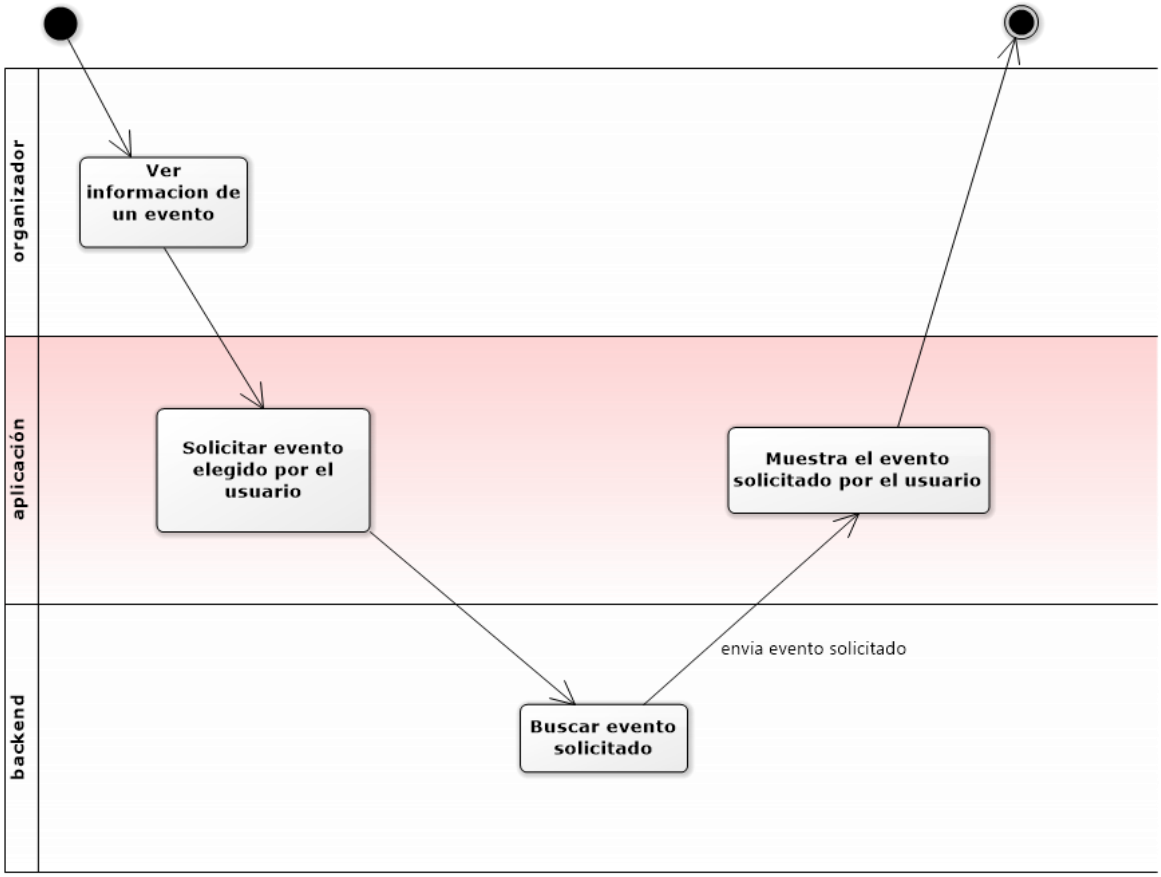
La figura 51 muestra las actividades para editar la contraseña de usuario. Se planteó dejar aparte del diagrama anterior para tener más control de la validación de las contraseñas.

Actividad - Crear evento



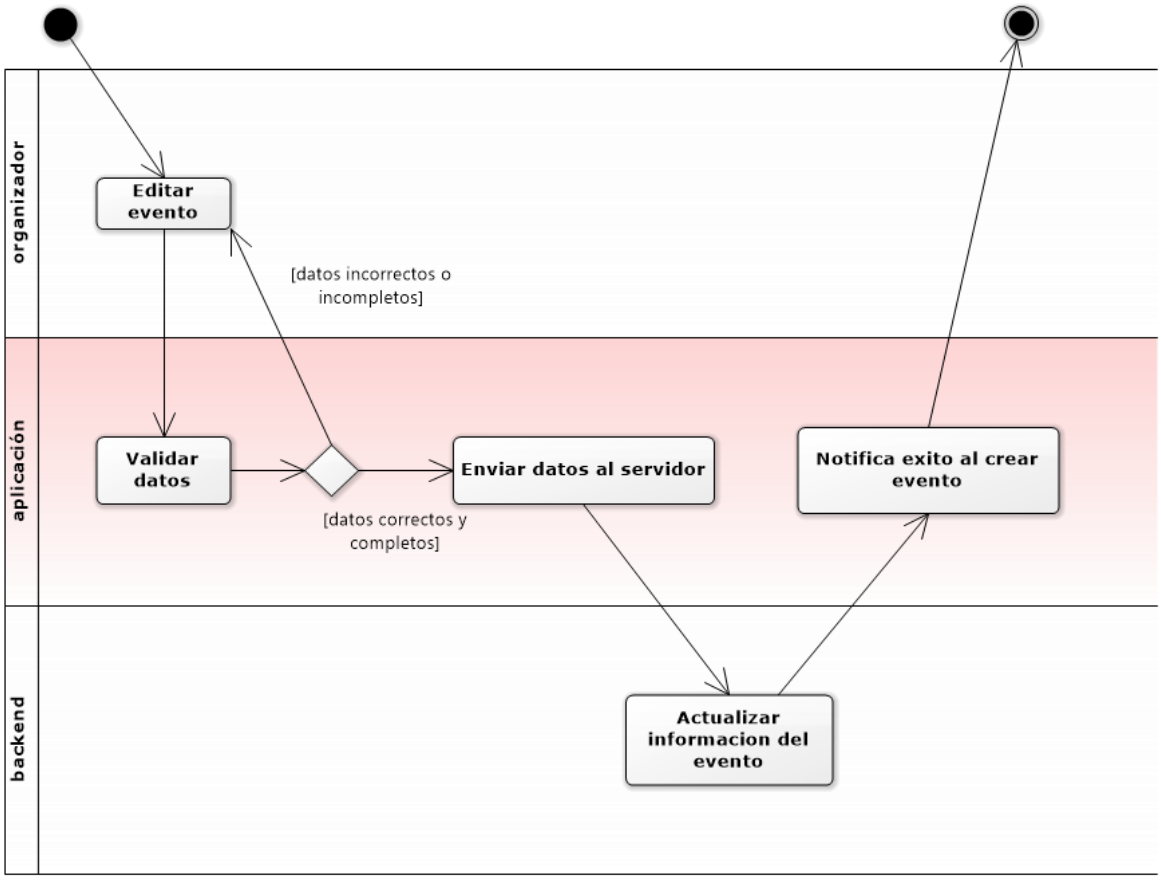
Cual el usuario desea crear un evento primero debe completar la información del formulario de creación de evento. Una vez completado y validado la información, estos datos se envían al servidor, allí se guarda la información y se devuelve la confirmación al usuario junto con el identificador del nuevo evento. De vuelta a la aplicación, esta valida si existen imágenes por cargar, si así es, primero valida que las imágenes cumplan con los requisitos de peso y cantidad establecidos. En caso afirmativo envía las imágenes junto con el identificador del evento que está siendo creado. El servidor guarda las imágenes, crea los respectivos datos relacionados a las imágenes y estos a su vez se asocian con el evento que está siendo creado. Una vez finalizado el servidor retorna la información completa del evento, de manera que la aplicación puede visualizar también las imágenes cargadas.

Actividad - Ver evento



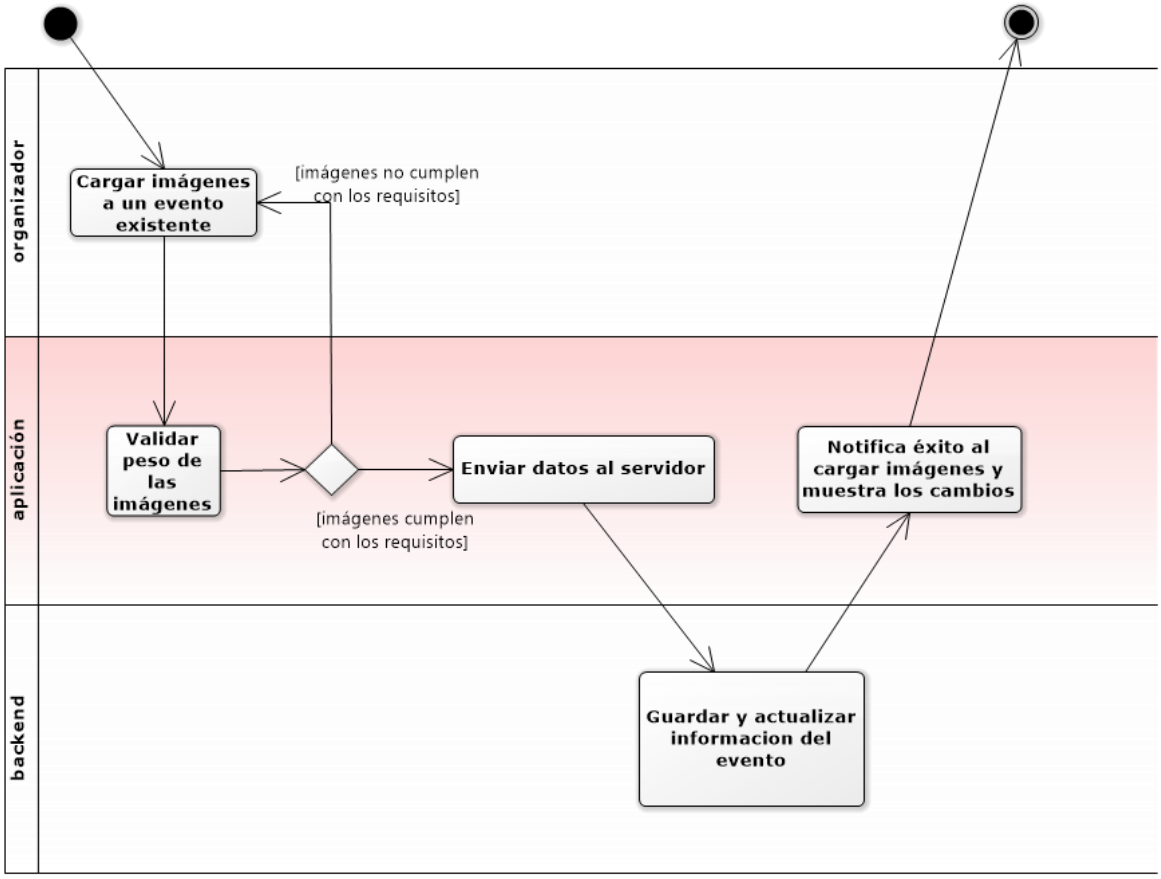
Para ver un evento el usuario selecciona un evento en la aplicación, esta a su vez solicita este recurso al servidor. El servidor responde con toda la información del evento y la aplicación muestra al usuario esta información.

Actividad - Editar evento



Similar a la figura 53, el usuario cuando visualiza el evento, puede modificar su información. Si así lo hace, de nuevo esta información nueva es validada por la aplicación, de ser correcta, se envían los datos al servidor. Este actualiza los cambios y devuelve la confirmación respectiva.

Actividad - Cargar imágenes a un evento



El usuario que ve la información de un evento puede decidir solo modificar las imágenes de dicho evento. En ese caso, las imágenes deben ser validadas si cumplen con el peso y cantidad establecidos. Si así ocurre, se envían al servidor. Este guarda las imágenes y genera información asociada a las imágenes guardada, las relaciona al evento modificado y finalmente devuelve información completa del evento junto con su confirmación de éxito.